LEARNING TASK REQUIREMENTS FOR COALITION FORMATION IN HETEROGENEOUS MULTI-AGENT SYSTEMS

A Thesis
Presented to
The Academic Faculty

By

Anusha Srikanthan

In Partial Fulfillment
of the Requirements for the Degree
of Master of Science in the
School of Electrical and Computer Engineering
Department of Electrical and Computer Engineering

Georgia Institute of Technology

August 2021

LEARNING TASK REQUIREMENTS FOR COALITION FORMATION IN HETEROGENEOUS MULTI-AGENT SYSTEMS

Thesis committee:

Dr. Harish Ravichandar
School of Interactive Computing
Georgia Institute of Technology

Dr. Sonia Chernova
School of Interactive Computing
Georgia Institute of Technology

Dr. Matthew Gombolay
School of Interactive Computing
Georgia Institute of Technology

Dr. Matthieu Bloch
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date approved: July 23, 2021

You will be the one to face the consequences of the decisions you make

My dad

For my grandmother, who always believed in me

ACKNOWLEDGMENTS

I would like to thank the members of my thesis committee for their help in preparation of this work – Dr. Harish Ravichandar, who is my biggest inspiration and without whom I would have been doomed to never complete it, Dr. Sonia Chernova, who helped to shed new light on many of my ideas, Dr. Matthew Gombolay, who gave me confidence with his great feedback on my first paper submission, and Dr. Matthieu Bloch, who always had golden advice.

Special thanks are due to my roommate, Ashwini and my friends from Georgia Tech who made this work possible. My colleagues at RAIL lab were invaluable both as friends and as sounding boards for some of my more outlandish ideas. I am also grateful to my best friend, Gautham, who was always there for me when I had bouts of second thoughts about doing research. Lastly, I would like to thank my parents who have been supportive throughout this arduous journey.

The author gratefully acknowledges the support for this work offered by Army Research Laboratory under grant award number W911NF2020036.

Any views and conclusions contained herein are those of the author, and do not necessarily represent the official positions, express or implied, of the funders.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Existing approaches to coalition formation assume that requirements associated with tasks are precisely specified by the human operator. However, prior work in psychology has demonstrated that humans, while extremely adept at solving complex problems, struggle to explicitly state their solution strategy. This thesis contributes two frameworks to learn *implicit* task requirements directly from expert demonstrations of coalition formation.

In the first framework, we account for the fact that demonstrators may allocate different, equally-valid coalitions to the same task. We assume that each such coalition results in optimal task performance. Essentially, we contribute a framework to model and infer such heterogeneous strategies to coalition formation. Our framework includes a resource-aware approach to generalize the inferred strategies to new teams without requiring additional training. To this end, we formulate and solve a constrained optimization problem that *simultaneously* selects the most appropriate strategy for a given target team, and optimizes the constituents of its coalitions accordingly. We evaluate our approach against several baselines, including some that resemble existing approaches, using detailed numerical simulations, StarCraft II battles, and a multi-robot emergency-response scenario. Our results indicate that our framework consistently outperforms baseline approaches in terms of requirement satisfaction, resource utilization, and task success rates.

Our second framework relaxes the typical assumption that the available demonstrations are optimal and incorporates interactive learning. Prior work in Learning from Demonstrations (LfD) depend on high quality demonstrations and result in poor generalization performance. Further, LfD approaches only perform as well as the best example in the demonstration set. Deviating from prior work, we assume access to sub-optimal demonstrations and evaluations of the assigned teams in the form of task-wise scores. In order to effectively learn from sub-optimal demonstrations, our framework infers the underlying reward function and subsequently generates coalitions by optimizing the inferred reward

function. In addition to learning from sub-optimal demonstrations, we utilize interactions with the environment to fine tune the reward distribution and obtain an estimate of the requirements for tasks. Specifically, we develop a bandit-based approach that can deal with continuous action spaces, and can be bootstrapped by sub-optimal demonstrations. We evaluate our approach against baselines that are inspired from prior work using detailed numerical simulations. Our results show that our approach combining both passive and interactive learning achieves higher task performance when generalizing to new teams when compared to baseline approaches that are similar to imitation learning, or utilize passive learning or interactive learning in isolation.

CHAPTER 1
INTRODUCTION

In this thesis, we focus on the *coalition formation* problem in which a team of robots need to be divided or partitioned into non-overlapping sub-teams (i.e., coalitions) such that multiple concurrent tasks can be successfully carried out [1, 2]. In particular, we are interested in coalition formation for *heterogeneous teams* (i.e., teams made of robots with different capabilities). Forming effective coalitions with heterogeneous multi-robot systems has been proven to be effective in several domains, such as environmental monitoring [3], agriculture [4], and construction [5].

Most existing approaches to coalition formation assume that the task requirements associated with the different tasks are explicitly specified by the human operator (e.g., [6, 7]). However, prior work has demonstrated that manually specifying multi-dimensional objective functions that capture the trade-offs between multiple factors can be very challenging [8, 9, 10]. In fact, complex tasks involve multi-dimensional requirements (e.g., searching an area, carrying a certain weight, and navigating rugged terrain) that are hard to explicitly and precisely specify.

Despite the challenges in explicit specification of requirements and objectives, domain experts routinely solve complex problems with similar requirements [8]. For example, the field of Inverse Reinforcement Learning (IRL), which is based on the finding that users have an easier time *demonstrating* a task than specifying a reward function, has led to successful applications in autonomous flight [11], social navigation [12], and manipulation [13].

In the first part of this thesis, we take an IRL-inspired approach to coalition formation. Instead of requiring exact task requirements, we propose to leverage *expert demonstrations* of coalition formation. Our approach is similar in spirit to IRL in that we attempt to infer the task requirements that the expert was implicitly attempting to satisfy when providing

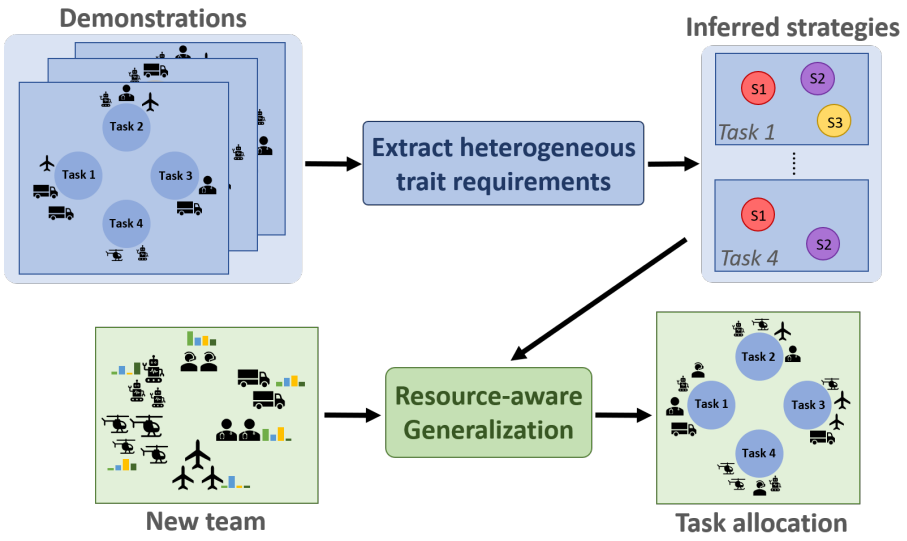


Figure 1.1: A block diagram of our framework to extract and generalize heterogeneous task requirements.

the demonstrations. Specifically, we define task requirements in terms of the *traits* (i.e., multi-dimensional capabilities) required to accomplish the task. We then formulate our problem as determining i) the *trait requirements* of each task from expert demonstrations, and ii) the *allocation* of robots to each task such that each coalition collectively satisfies the inferred requirements.

A key challenge in learning from human demonstrations is that different users may demonstrate different, equally-valid, solutions to a task. A rich body of work in psychology and human-robot teaming suggests that complex tasks are often solved using one of many comparable strategies [14, 15]. For instance, one can search an area either with i) a slow-moving coalition with a large collective sensing area, or with ii) a fast-moving coalition with a small collective sensing area. We denote such different, yet equivalent, trait requirements as *heterogeneous strategies*.

In essence, we contribute: 1) a novel formulation of heterogeneous coalition formation strategies based on trait requirements, 2) a clustering-based approach for inferring such generalizable heterogeneous strategies from expert demonstrations, and 3) an optimization-based method to resource-aware selection of strategies to generalize the inferred strategies

to entirely new target teams without additional training.

We propose a hierarchical clustering-based method to extract heterogeneous requirements from expert demonstrations consisting of successful assignments (see top half of Figure 1.1). The clustering approach provides appropriate *abstraction* of trait requirements by distilling expert demonstrations into a small number of strategies.

To enable *resource-aware* generalizations of the inferred strategies to new target teams, we pay explicit attention to the compatibility of the team's capabilities with the inferred strategies for each task (see bottom half of Figure 1.1). We propose an optimization-based approach that *simultaneously* i) selects the strategy for each task and ii) optimizes the constituents of the coalitions, both based on the *context* of the target team's capabilities and trait requirements.

Our framework accounts for task-interdependence by solving a constrained optimization problem to *simultaneously* optimize both strategy selection and robot assignment. This is important because the team has finite resources, and as such, the selection of strategy for one task affects the selection for other tasks.

In the second part of this thesis, our focus shifts to dealing with sub-optimal demonstrations. Few approaches in task allocation use demonstrations for learning to allocate robots (see [16] for an exception). This is mainly because collecting high quality demonstrations from experts is expensive [17]. Moreover, the demonstrations are assumed to be optimal for methods using imitation learning. Hence, imitation learning suffers from the problem of performing only as good as the best demonstration available within the dataset [17]. To address the limitations of using sub-optimal demonstrations, we assume access to task-wise scores obtained by the coalitions assigned by the demonstrations. We explore the possibility of learning to evaluate the team's performance on tasks using their respective scores.

Additionally, LfD approaches incur large generalization errors when the team performing the tasks are changed [17]. To overcome generalization errors, our approach to the

problem of learning from sub-optimal demonstrations is two-fold. First, we learn the underlying reward distribution using sub-optimal demonstrations and interactions from the environment. Second, we compute the task requirements in terms of the agent capabilities and solve a constrained optimization problem to find the task-coalition pairs.

For the first part of our approach, our objective is to learn to evaluate coalitions from sub-optimal demonstrations. The reward function for each task used for the evaluation of coalitions depends on the different aggregated capabilities assigned by the users to that task. We denote sub-optimality in terms of the task-wise scores obtained by the coalitions while performing the tasks. In order to learn the underlying reward distribution and estimate the task performance of coalitions, we introduce a bandit-based interactive learning framework.

To effectively generalize when new species are a part of the team, our bandit-based interactive learning framework computes arms in terms of the aggregated agent capabilities. As a result, the multi-armed bandit approach selects arms in continuous spaces. Hence, we model the reward distribution as a Gaussian process and design an algorithm inspired from prior work on the use of Gaussian process optimization in the bandits setting [18]. We make an active choice of selecting arms according to the Upper Confidence Bound selection rule.

In the second part of our approach, we select the arms achieving the highest score on each task as the task requirements. We solve a constrained least squares problem to form coalitions that meet the task requirements as closely as possible, constrained by the finite resources of our team.

We validate our contributions by evaluating our approach against baselines using detailed numerical simulations. We select baselines that are similar to methods in imitation learning, passive reward learning and interactive learning. Our results show evidence that our approach of combining interaction-based learning with sub-optimal demonstrations performs the best in terms average generalization performance and achieving the highest scores on tasks.

CHAPTER 2
BACKGROUND

In this chapter, we establish the background of the thesis by discussing related work for both resource-aware generalization and learning from sub-optimal demonstrations, for coalition formation. We divide prior work into the following broad categories to contextualize our contributions.

2.1 Categories in Task Allocation

Formally, task allocation problems are categorized based on three axes: Single-Task (ST) vs. Multi-Task (MT) robots; Single-Robot (SR) vs. Multi-Robot (MR) tasks; and Instantaneous Allocation (IA) vs. Time-extended Allocation (TA). We refer readers to Korsah et al. [2] for a comprehensive categorization survey. Our work addresses the coalition formation problem, which is an instance of the ST-MR-IA problem.

We note that the coalition formation problem is closely related to the problems of multi-agent planning [19] and scheduling [20]. However, each of these problems differ considerably in terms of their objectives and what is assumed to be given. Specifically, planning approaches assume access to domain definitions and action models, while scheduling approaches assume access to the known set of actions to take. In contrast, coalition formation approaches do not assume access to such information and focus on optimizing the constituents of each coalition such that task requirements are satisfied. As such, we view our work as complementary to existing approaches for planning and scheduling.

2.2 Coalition formation

Existing approaches to coalition formation fall into one of three groups: utility-based, auction-based, and trait-based methods. First, prior work has demonstrated that utility-based methods can compute optimal coalitions based on the coalition values for each team assigned to a task [21, 22, 23, 24]. However, the design of utility functions requires considerable domain knowledge and often needs to be hand-crafted for the particular set of tasks and robots. Second, auction-based approaches have had tremendous success in solving a wide range of coalition formation problems using mechanisms through which robots bid on tasks [25, 26, 27]. A common limitation of auction methods is that they rely on extensive communication for bidding and scale poorly with the number of robots in the team. Lastly, recent advances in task allocation have introduced methods that model both robots and tasks in terms of their traits (i.e., capabilities) [6, 7]. We adopt a similar approach and model our robots and tasks in terms of traits. A unique benefit of using trait-based approaches is that they are generalizable to new teams, as task requirements are specified in terms of traits, and not specific robots.

We deviate from prior work in coalition formation in terms of two crucial assumptions. First, methods discussed thus far assume complete knowledge of task requirements, whatever the form. Second, most assume that there is exactly one set of task requirements (i.e., no heterogeneity). In contrast, we introduce a model for heterogeneous task requirements and contribute an algorithm that extracts such implicit requirements directly from expert demonstrations. Further, our approach satisfies task requirements by jointly optimizing strategy selection and robot assignment.

2.3 Heterogeneous learning from demonstrations

Next, we examine related work focused on learning from heterogeneous demonstrations. Recent advances in learning from demonstrations has introduced methods to capture het-

erogeneity from expert demonstrations. Prior work has showcased the ability to infer discrete modes of operation [28], multiple visual intentions [29], and most commonly diverse user preferences [30, 31, 32, 33] from heterogeneous demonstrations. Existing approaches that embrace heterogeneity in demonstrations focus on inferring and encoding the distinct characteristics. In contrast, our approach focuses on optimizing the selection of the most appropriate inferred strategy based on the available resources. Further, our work represents the first attempt to learn heterogeneous task allocation strategies from demonstrations.

2.4 Learning for task allocation

Additionally, we discuss attempts in the general area of multi-agent learning [34] and specifically scrutinize the details of learning methods for task allocation. Much of the literature on learning for multi-agent systems is focused on reinforcement learning applied to low-level coordination (e.g., [35, 36]). However, only a handful of methods exist for learning high-level coordination from demonstrations. Here, we discuss a few notable examples. An approach introduced in [16] models task features and learns their weights from expert demonstrations to bias the prices in a market-based task allocation algorithm. A recent structured prediction approach [37] for task allocation uses a combination of reinforcement learning and quadratic integer programming for learning directly from data to optimize assignments. The approaches in [16, 37], however, assume the existence of a single strategy for task allocation. A distributed approach for multi-agent task allocation [38] learns to select the most appropriate of two pre-specified strategies, namely Earliest Deadline First (EDF) or Nearest Task First (NTF). Finally, reinforcement learning has been utilized to address repeated coalition formation problems that involve teammates whose capabilities are initially unknown [39].

The learning methods discussed so far consider only homogeneous robots [37], do not show generalization to teams unseen during training [16, 40], depend on the ability to interact with the environment to learn policies [39] or adhere to a limited number of pre-

specified strategies [38]. In contrast, our framework is capable of learning generalizable and heterogeneous strategies for task allocation in heterogeneous multi-agent systems from expert demonstrations, and do not rely on environmental interactions.

2.5 Learning from sub-optimal demonstrations

For the second part of the thesis, we discuss prior work in learning from sub-optimal demonstrations. Many papers [41, 42, 43, 44] use human demonstrations of varying quality to improve the performance of reinforcement learning policies. Generally, most prior work in learning from sub-optimal demonstrations fall into two categories, i) contributes approaches to improve reinforcement learning policies using human demonstrations [41, 42, 43, 44] or ii) contributes approaches to improve imitation learning policies [45] by bootstrapping interactions with the environment. In [41], they show that learning from multiple teachers leads to the best performance and the authors of [44] use demonstrations to constrain exploration leading to the best success rates and sample efficiency. More recent advances [46, 33] in this field addressed trajectory optimization by using inverse reinforcement learning based approaches to learn the underlying reward distribution which is then used as the reward function for training RL agents. However, there has been no prior work to our knowledge for learning from sub-optimal demonstrations specifically, in task allocation. An important difference between prior work and learning from sub-optimal demonstrations in task allocation is that, the Markovian assumption to design Markov Decision Processes (MDPs) necessary for the reinforcement learning framework, does not hold. In our work, we take inspiration from prior work to bootstrap interaction-based learning with sub-optimal demonstrations from users, by using a bandit-based framework which does not require a Markovian assumption.

2.6 Interaction-based learning for Multi-agent Systems

Finally, we discuss related work that employ interaction-based learning such as bandit approaches to multi-agent systems. Multi-agent task assignment in the bandit framework [47] proposed the formulation of task allocation as a restless bandits problem with switching costs and discounted rewards. Their assumption that the sites to be serviced evolve as a Markov chain independently with different transition probabilities aid in designing MDPs. More recent work [48] addresses issues in crowdsourcing systems by estimating a worker's ability over time in a multi-armed bandit set up to improve the performance on allocated tasks. Further, two papers [49, 50] in multi-agent network optimization use local interactions among agents for improving the network performance. In [49], the authors formulate a multi-player multi-armed bandit approach to maximize the global reward while the authors of [50] use evolutionary task assignment approach for resource allocation. In our work, we address a key issue missing from most prior work, the problem of generalizing to new types of agents in coalition formation. To this end, we use traits [7], to both represent the capabilities of agents and an approximation for the requirements of tasks. Dealing with the problem in the trait space makes it more challenging because of it's continuous nature, however, gives us the benefit of generalization to new teams without any additional training. Hence, we build on the trait-based framework to propose a novel bandit-based interaction learning architecture and solve our problem using a Gaussian process multi-arm bandit approach inspired from [18].

CHAPTER 3

PRELIMINARIES

We consider a heterogeneous team composed on $S \in \mathbb{N}$ *species* (i.e., robot types), in which the s^{th} species contains $N_s \in \mathbb{N}$ robots. Let the team be tasked with a set of M concurrent tasks denoted by $\mathscr{T} = \{T_1, T_2, .., T_M\}$. We now introduce a series of pertinent definitions.

Definition 3.0.1 *The **capabilities** of the team are encoded by its Species-Trait matrix* $Q = [q_1, \cdots q_S]^T \in \mathbb{R}_+^{S \times U}$*, where* $q_s \in \mathbb{R}_+^U$ *is the trait vector listing the different traits of the* s^{th} *species, and* U *is the total number of traits.*

Definition 3.0.2 *The **assignment** of robots to tasks is encoded by the Assignment matrix* $X = [x_{ms}] \in \mathbb{Z}_+^{MxS}$*, where the* ms^{th} *element* x_{ms} *denotes the number of robots from Species s assigned to Task* T_m*.*

Definition 3.0.3 *The **aggregated traits** available for each task due to the assignment of robots is encoded in the Task-Trait matrix* $Y \in \mathbb{R}_+^{M \times U}$*, whose* m^{th} *row represents the aggregation of traits for Task* T_m *and is given by*

$$y_m = Q^T x_m, \ \forall m \in \{1, \cdots, M\} \tag{3.1}$$

Definition 3.0.4 *Let a set of heterogeneous **strategies** associated with the* m^{th} *task be made up of* P_m *strategies. Each strategy denotes the trait requirements, that when satisfied, leads to successful completion of the task. As such, the* r^{th} *strategy for the* m^{th} *task is given by*

$$^r y_m^* \in \mathbb{R}_+^U, \forall r \in \{1, \cdots, P_m\} \tag{3.2}$$

Note that our definition of strategies allows for different sets of requirements to be associated with the same task. We assume that satisfying any one of the P_m task requirements

will result in successful completion of the m^{th} task.

Definition 3.0.5 *Let a set of N expert **demonstrations** in the form of robot assignments be given by*

$$\mathscr{D} = \{X^{(i)},\ Q^{(i)}\}_{i=1}^{N} \tag{3.3}$$

where $X^{(i)}$ represents the expert-specified assignment matrix for a team with capabilities encoded by the Species-Trait matrix $Q^{(i)}$.

Given the above definitions, our initial problem consists of two steps: *i)* extracting the set of approximated strategies ${}^{r}\hat{y}_m$ for each task from $\mathscr{D}$, and *ii)* optimizing the Assignment matrix $X^{(j)}$ of a new team with Species-Trait matrix $Q^{(j)} \notin \mathscr{D}$ such that the associated $y_m \succ {}^{r}\hat{y}_m,\ \forall m$, where y_m denotes the traits aggregated by the target team towards the m^{th} task and $\succ$ denotes the element-wise greater-than operator.

In the second part of this thesis, we tackle the question of coalition formation when the demonstrations are sub-optimal with respect to team performance. The team performance is measured by the task-wise scores obtained by the allocated coalitions. We assume access to the task-wise scores as part of the demonstration set. To be precise, we define the task-wise reward function as follows.

Definition 3.0.6 *The **reward function** for task m is defined as a mapping from aggregated traits to scores, which are the ground truth rewards measuring the quality of the allocated coalitions.*

$$r_m : \mathscr{Y} \rightarrow \mathbb{R}^{+},\ \forall m \tag{3.4}$$

where r_m represents the mapping function and $\mathscr{Y}$ represents the space of all possible aggregated trait vectors.

Definition 3.0.7 *The **score** for task m is the output of the reward function r_m for a particular x_m and Q, which measures the quality of the allocated coalition on that task.*

$$s_m = r_m(Q^T x_m), \forall m \tag{3.5}$$

where s_m represents the ground truth reward. The vector representing the scores for all tasks in $\mathscr{T}$ is given by $S \in \mathbb{R}_+^M$.

We now define the demonstration set consisting of sub-optimal teams.

Definition 3.0.8 *Let a set of $N_{sub} \in \mathbb{N}$* ***sub-optimal demonstrations*** *in the form of agent assignments be given by*

$$\mathscr{D}_s = \{X^{(k)},\ Q^{(k)},\ S^{(k)}\}_{k=1}^{N_d} \tag{3.6}$$

where $X^{(k)}$ represents the user-specified assignment matrix for a team with capabilities encoded by the Species-Trait matrix $Q^{(k)}$ and $S^{(k)} \preceq S_{max},\ S^{(k)} \in \mathbb{R}_+^M$ denotes the scores for M tasks. The $\preceq$ is the element-wise less than or equal to operator indicating that the scores obtained by the demonstrations are lesser than the maximum scores attainable represented by $S_{max} \in \mathbb{R}_+^M$.

CHAPTER 4
RESOURCE-AWARE GENERALIZATION OF HETEROGENEOUS STRATEGIES

In this section, we provide details of our proposed framework. We peruse a running example to illustrate each component of our framework. To this end, let us consider a disaster response mission involving tasks such as removing debris, searching urban environments, and retrieving objects or people from damaged buildings.

We validate our contributions by evaluating our approach against a variety of baselines [7, 22, 16, 40], using detailed numerical experiments, battles in StarCraft II game, and a disaster response scenario on the Robotarium [51]. Our results demonstrate that our approach consistently outperforms the baseline approaches in terms of requirement satisfaction, resource utilization, and task success rates.

4.1 Problem Statement

Given a set of demonstrations $\mathscr{D}$ for the task set $\mathscr{T}$, we are interested in *i)* extracting the set of approximated strategies ${}^{r}\hat{y}_m$ for each task, and *ii)* optimizing the assignment matrix $X^{(j)}$ for a new team whose species-trait distribution is given by $Q^{(j)} \notin \mathscr{D}$.

4.2 Extracting heterogeneous task requirements

In order to extract implicit task requirements, we first compute the trait aggregations from the demonstrations $\mathscr{D}$ as

$$y_m^{(i)} = {Q^{(i)}}^T x_m^{(i)}, \quad \forall m, i \tag{4.1}$$

where $y_m^{(i)}$ represents trait aggregation associated with the m^{th} task of the i^{th} demonstration. Given that there might be multiple strategies to solve each task, the computed trait aggre-

gations in (Equation 4.1) are likely to form distinct clusters in the trait space, with each cluster representing a unique set of trait requirements or strategy.

Note that we do not attempt to capture the specific robots (e.g., 5 quadrupeds and 3 aerial vehicles) used by an expert. Instead, we extract the amounts of specific traits that are possessed by the experts' coalitions (e.g., 1 km^2 aggregate coverage area, and 5 robots with ability to navigate rugged terrain). Adopting these trait-based specifications allows us to generalize the extracted requirements to new teams containing entirely new kinds of robots.

We then apply agglomerative (i.e., hierarchical bottom-up) clustering on the computed trait aggregation vectors for each task to obtain clusters denoted by $\{{}^{r}C_m\}_{r=1}^{P_m}$. While our framework is agnostic to the specific clustering approach, hierarchical clustering allows for adaptively choosing the number of clusters based on the task's characteristics. On a related note, the choice of the neighborhood distance thresholds can be made from analyzing the dendrogram plot for each task.

Once the clusters are identified, we compute the approximate task requirements associated with each strategy of each task as follows

$$ {}^{r}\hat{y}_m = \sum_{y_m^{(i)} \in {}^{r}C_m} \frac{y_m^{(i)}}{|{}^{r}C_m|} \tag{4.2} $$

where $|{}^{r}C_m|$ denotes the number of demonstrations for Task T_m that are identified as part of the r^{th} cluster. Utilizing the average trait aggregation as an approximation of the task requirements associated with each strategy helps deal with potential noise in the demonstrations.

Our notion of heterogeneity in strategies does not point to the fact that different coalitions can carry out the task in a similar fashion (e.g., helicopters or fixed-wing aircraft can be used to aerially search an area). Rather, we reserve the term "strategies" to denote fundamentally different, yet equivalent, approaches to carrying out the task (search the area

aerially vs. from the ground).

Revisiting our disaster response example from before, we could extract two distinct strategies from multiple expert coalitions assigned to the search task: one requiring large sensing radius and slow speed, and another requiring small sensing radius and high speed.

4.3 Resource-aware coalition formation

Given a new target team with the Species-Trait matrix $Q^{(j)} \notin \mathscr{D}$, we turn to the problem of optimizing its coalitions such that the inferred task requirements are satisfied. To this end, we formulate a constrained optimization problem capable of *simultaneously i)* choosing strategies for each of the M tasks based on the resources available to the team, and *ii)* optimizing robot assignment such that the requirements associated with the chosen strategies are met.

To represent the choice between multiple strategies, we introduce the *strategy selectors* $z_m \in \{0,1\}^{P_m}, \forall m$, one-hot encoded decision variables that identify the choice of strategy for each task. In addition, we introduce integer decision variables $x_m \in \mathbb{Z}_+^S, \forall m$ representing the assignment of robots to each task. We treat each task as a sub-problem and simultaneously optimize the overall assignment such that the chosen set of trait requirements are satisfied for all tasks.

To quantify trait satisfaction, we define an error measure called trait mismatch error as follows. If the r^{th} strategy is chosen for Task T_m, the corresponding Trait Mismatch Error ${}^r e_m$, between the aggregated traits and the trait requirements of the r^{th} strategy is given by

$$ {}^r e_m = \| {}^r \hat{y}_m - {Q^{(j)}}^T x_m \|_2^2 \tag{4.3} $$

where x_m represents the assignment for the Task T_m.

Based on the definitions above, we define the net trait mismatch error for Task T_m as

$$E_m = z_m^T \, e_m \tag{4.4}$$

where $e_m = [{}^1e_m, \cdots, {}^{P_m}e_m]^T \in \mathbb{R}_+^{P_m}$ is a vector of trait mismatch errors between the aggregated traits and each of the strategies' task requirements.

Finally, we cast the *resource-aware* optimization of robot assignments for the new team with $Q^{(j)} \notin \mathscr{D}$ as a constrained quadratic integer program:

$$\{x_m^{*(j)}, z_m^{*(j)}\}_{m=1}^{M} = \arg\min_{x_m, z_m} \sum_m E_m \tag{4.5}$$

$$\text{s.t.} \sum_m x_m \preceq N_a \tag{4.6}$$

$$z_m^T \cdot \mathbf{1} = 1, \; \forall m \tag{4.7}$$

$$\hat{Y}_m z_m \preceq {Q^{(j)}}^T x_m, \; \forall m \tag{4.8}$$

where $N_a \in \mathbb{Z}_+^S$ represents the vector of total robots per species, $\mathbf{1}$ is a vector of ones, and $\hat{Y}_m = [{}^1\hat{y}_m, \cdots, {}^{P_m}\hat{y}_m] \in \mathbb{R}_+^{U \times P_m}$ represents all the distinct trait requirements for Task T_m extracted from the demonstrations.

Our approach is resource-aware since it optimizes both task allocation and strategy selection while taking into account the resources available to the target team as given by $Q^{(j)}$. As shown in (Equation 4.6), we explicitly constrain our approach to not recruit more robots than available in each species. The constraint in (Equation 4.7) ensures that only one strategy is chosen for each task. Finally, the constraint in (Equation 4.8) ensures that the assignment satisfies the desired trait requirements associated with the chosen strategies. However, a notable limitation of our approach is that the optimization problem above is not guaranteed to converge to the global optimum.

In scenarios with under-resourced teams, the optimization problem defined in (Equation 4.5)-(Equation 4.8) might be infeasible. In such cases, we relax the problem by remov-

Algorithm 1: Resource-aware coalition formation

Input : $\{^{r}\hat{y}_m\}, \forall r, m$, and $Q^{(j)} \notin \mathscr{D}$
Output: $X^{*(j)}, Z^{*(j)}$

Initialize int x_m and z_m, $\forall m \in \{1, \cdots, M\}$
try:
    /* Attempt to satisfy all trait requirements */
    Compute trait mismatch for each task from (Equation 4.4)
    Apply constraints (Equation 4.7), (Equation 4.6), and (Equation 4.8)
    Optimize cost in (Equation 4.5) subject to (Equation 4.7), (Equation 4.6), and (Equation 4.8)
except:
    /* If under-resourced, relax minimum requirements */
    Compute trait mismatch for each task from (Equation 4.4)
    Apply constraints (Equation 4.7) and (Equation 4.6)
    Optimize cost in (Equation 4.5), subject to (Equation 4.7) and (Equation 4.6)
return $X^{*(j)}, Z^{*(j)}$

ing the constraints in (Equation 4.8) as described in algorithm 1. This relaxation allows our approach to adaptively choose strategies and minimize the trait mismatch error for under-resourced teams.

Revisiting our disaster response example, our framework chooses between the two strategies for the search task (low speed and large sensing radius vs. high speed and small sensing radius) depending on the capabilities of the available team. Indeed, one of the two strategies is likely to be better suited than the other for a given target team. Further, the resource constraint in (Equation 4.6) helps our framework realize that if all ground vehicles are assigned to the search task, we will not be able to utilize them to remove debris.

4.4 Experimental Evaluation

We evaluate our approach [1] in terms of its ability to satisfy trait requirements when forming coalitions for new teams using detailed numerical simulations, StarCraft II battles, and a multi-robot emergency-response scenario. Specifically, we evaluate our framework's ability to generalize the inferred heterogeneous task allocation strategies to teams with different

[1] https://github.com/Ravichandar-Lab/Resource-Aware-Generalization

number of robots, and different types of species.

4.4.1 Baselines

In all the experiments below, we compare the performance of our approach against the following baselines that take different factors into account:

1. No Heterogeneity (NH): This baseline assumes the existence of a single set of task requirements. For this baseline, we compute the trait requirements as an average across all demonstrations for each task:

$$y^*_{m_{uni}} = \frac{\sum_{j=1}^{N} y_m^{(j)}}{N} \tag{4.9}$$

With only one strategy, task allocation is optimized by solving a constrained least squares problem. We included this baseline to validate the need to consider the heterogeneity in task requirements.

2. No Context (NC): This baseline extracts multiple strategies from demonstrations like our approach. However, when optimizing for assignments, it picks strategies at random irrespective of the team's capabilities (i.e., the context) and solves a constrained least squares problem. We included this baseline to validate the need to consider the team's resources when selecting which strategy to pursue.

3. No Abstraction (NA): This baseline does not cluster the demonstrations and solves the optimization problem in (Equation 4.5)-(Equation 4.8) with every demonstration as a potential strategy. We included this baseline to validate the need to distill the demonstrations into a small number of strategies.

4. Random Allocation (RA): This baseline entirely ignores the notion of task requirements and assigns a random subset of the available robots to each task without overlap. We included this baseline to validate the need for careful task allocation.

We note that both NH and NC baselines resemble prior work in coalition formation (e.g., [22, 16, 40, 7]) as they rely on a single pre-specified set of task requirements. In contrast to all baselines, our approach embodies all three considerations (heterogeneity, context,

and abstraction) and solves the optimization problem in (Equation 4.5)-(Equation 4.8) after inferring the strategies from the demonstrations.

4.4.2 Metrics

We evaluate each allocation using the following metrics:

1. Minimum trait mismatch: is defined as the smallest difference between the achieved trait aggregation and any of the potential task requirements, if the achieved trait aggregation is lesser than all the potential task requirements, otherwise it is 0. In other words, minimum trait mismatch does not penalize over-provisioning and is given by,

$$E_{\min}(\hat{x}_m) = \begin{cases} \frac{\min_r(\|{}^r\hat{y}_m - Q^T\hat{x}_m\|_2)}{\|{}^r\hat{y}_m\|_2} & \text{if } {}^r\hat{y}_m \succ Q^T\hat{x}_m \\ 0 & \text{Otherwise} \end{cases} \tag{4.10}$$

where $\hat{x}_m$ denotes the assignment for Task T_m computed by an approach, and Q represents the species-trait matrix of the target team.

2. Exact trait mismatch: is defined as the smallest difference between the achieved trait aggregation and any of the potential task requirements, given by

$$E_{\text{exact}}(\hat{x}_m) = \frac{\min_r(\|{}^r\hat{y}_m - Q^T\hat{x}_m\|_2)}{\|{}^r\hat{y}_m\|_2} \tag{4.11}$$

We compute the smallest difference since all strategies are equivalent and it is sufficient to satisfy the requirements associated with any one of them.

3. Robot utilization: In addition to trait mismatch errors, we also define a metric to account for the relative number of robots recruited by each algorithm as

$$RU(\hat{x}_m) = \frac{\hat{x}_m^T \cdot \mathbf{1}}{N_a \cdot \mathbf{1}} \tag{4.12}$$

4. Success Rate: Finally, we measure the team's performance in experiments from subsec-

tion 4.4.4 and subsection 4.4.5, based on their success or failure in executing the tasks:

$$SR(\hat{x}_m) = \frac{N_{sucess}(\hat{x}_m)}{N_{runs}} \times 100 \tag{4.13}$$

where $N_{success}(\hat{x}_m)$ denotes the number of times the tasks were successfully executed out of N_{runs} attempts. We also report the computational costs associated with each approach in Appendix D.

Combined, these metrics measure the ability of an approach to effectively optimize the allocation such that the corresponding task requirements are met while recruiting the least number of robots. We statistically analyze all our results using the Kruskal-Wallis test, followed by the Dunn test for post-hoc pairwise comparisons and FDR adjustment.

4.4.3 Numerical analyses

We first evaluate all the approaches from subsection 4.4.1 using the metrics from subsection 4.4.2 when choosing the coalitions for simulated target teams that were not encountered in the demonstrations. These detailed numerical simulations help analyze the performance of each algorithm across a wide variety of problems by altering aspects such as team size, robot capabilities, and task strategies. See Appendix A for additional experiments.

Design

We consider simulated task allocation problems, each with four species ($S = 4$), three traits ($U = 3$), three tasks ($M = 3$), three strategies per task ($P_m = 3, \forall m$), and the number of robots per species uniformly randomly sampled between 6 and 33. We generate a set of 240 demonstrations ($\mathscr{D} = \{X^{(i)}, Q^{(i)}\}_{i=1}^{240}$) and test our approach and the baselines on 60 target teams ($\{Q^{(j)}\}_{j=241}^{300}$). We generate teams and strategies such that our data contains a mix of under-, sufficiently-, and over-resourced teams. We set 1800 seconds as an upper bound on the optimization time for all approaches.

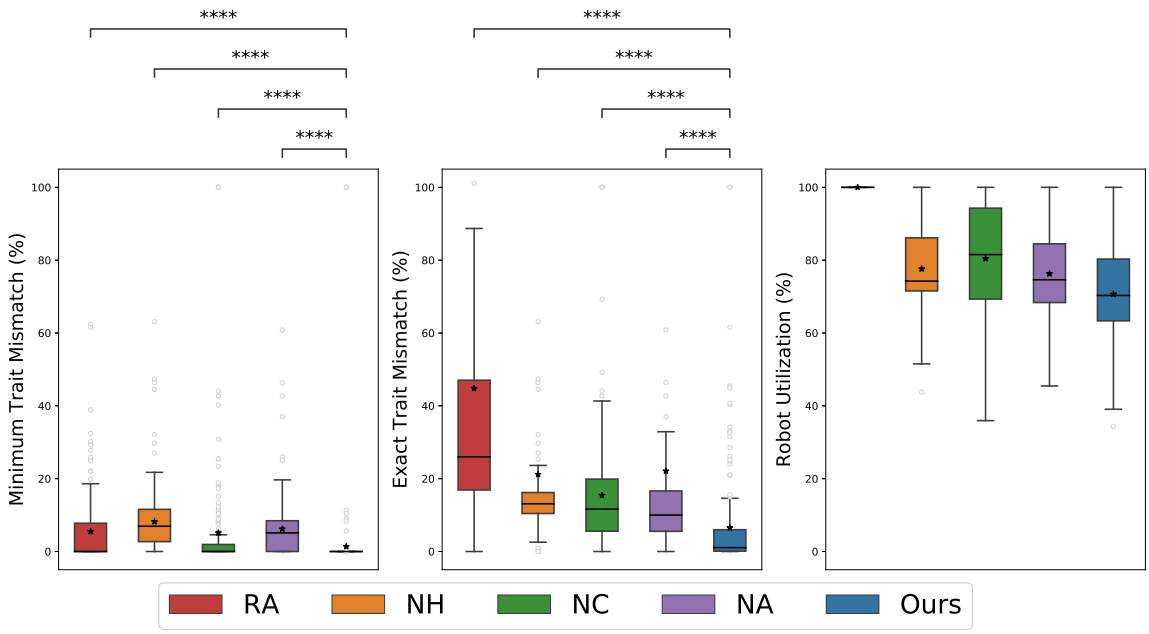


Figure 4.1: Measures of trait satisfaction and robot utilization percentages on numerical experiments (lower is better).

Results and discussion

First, our approach was able to extract three distinct strategies for each task from the demonstration set $\mathscr{D}$. Given the inferred strategies, we evaluated each algorithm's ability to optimize the coalitions of the target team in terms of the metrics in subsection 4.4.2. As shown in Figure 4.1, the RA baseline consistently recruits all the available robots and yet results in the highest E_{exact} and a non-trivial E_{min}. In contrast, our approach (ours) and all other baselines (NH, NC, and NA) utilize fewer proportion of robots and result in comparable or lower errors. These observations demonstrate the need for careful task allocation.

We find that our approach (ours) outperforms all the baselines (NH, NC, and NA) in terms of both minimum and exact trait mismatch errors, and that the improvements are statistically significant ($p < 1e-5$). Further, our approach is able to do so while utilizing a comparable number of robots. This observation indicates that our approach utilizes the recruited robots more effectively than all the baselines. We also observe that ignoring heterogeneity (NH) leads to poorer trait satisfaction as indicated by higher E_{min}, and

Figure 4.2: Snapshot of our StarCraft II battle simulation designed on a map inspired from [52].

failing to consider the team's resources (NC) results in over-provisioning as indicated by higher E_{exact}. This observation points to the deficiencies of existing coalition formation approaches that either ignore heterogeneous strategies or the context of available resources.

Finally, ignoring abstraction (NA) incurred a considerably larger computational cost (1067.67 ± 441.47s) than our approach (11.12 ± 26.94s)[2]. This increased computational burden of NA is a natural result of considering all data points in $\mathscr{D}$ as a unique strategy. In contrast, our approach (ours) reasons over a smaller number of distinct strategies that are distilled from the demonstrations.

4.4.4 Evaluation on StarCraft II

In the next set of experiments, we evaluate all the approaches from subsection 4.4.1 on StarCraft II battles using the metrics from subsection 4.4.2. The agents in the game belong

[2]See Appendix Appendix D for more details

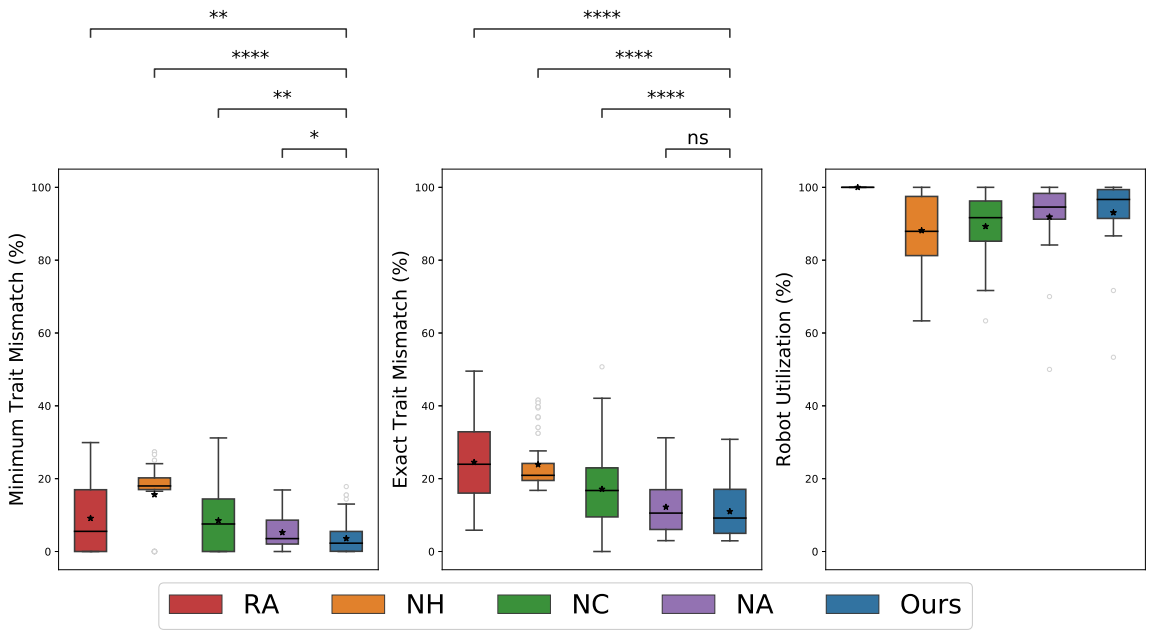


Figure 4.3: Measures of trait satisfaction and robot utilization on StarCraft battles (lower is better).

to different species, each with a particular set of capabilities as listed in Table B.1 in Appendix B. We handle trajectory-level control of the agents by using in-built AI bots from StarCraft II editor to ensure that, differences in their individual actions do not interfere with our assessment of the coalitions' performance. The battles emulate tasks that require careful allocation using combinations of the available species. The battles also aid in the evaluation of our approach against the baselines in terms of task-level performance. To compare the performance of all approaches, we compute the percentage success rate SR as defined in (Equation 4.13).

Design

We design simulated battles based on the 2s3z map from the StarCraft II Multi-Agent Challenges (SMAC) [52]. Specifically, we define four concurrent tasks ($M = 4$) on the battle map, each involving a battle with 10 Stalkers and 15 Zealots.

We generate 9 demonstrations ($\mathscr{D} = \{X^{(i)}, Q^{(i)}\}_{i=1}^{9}$) representing three strategies ($P_m = 3$) (as shown in Table B.2), each involving four different species ($S = 4$), for the battle

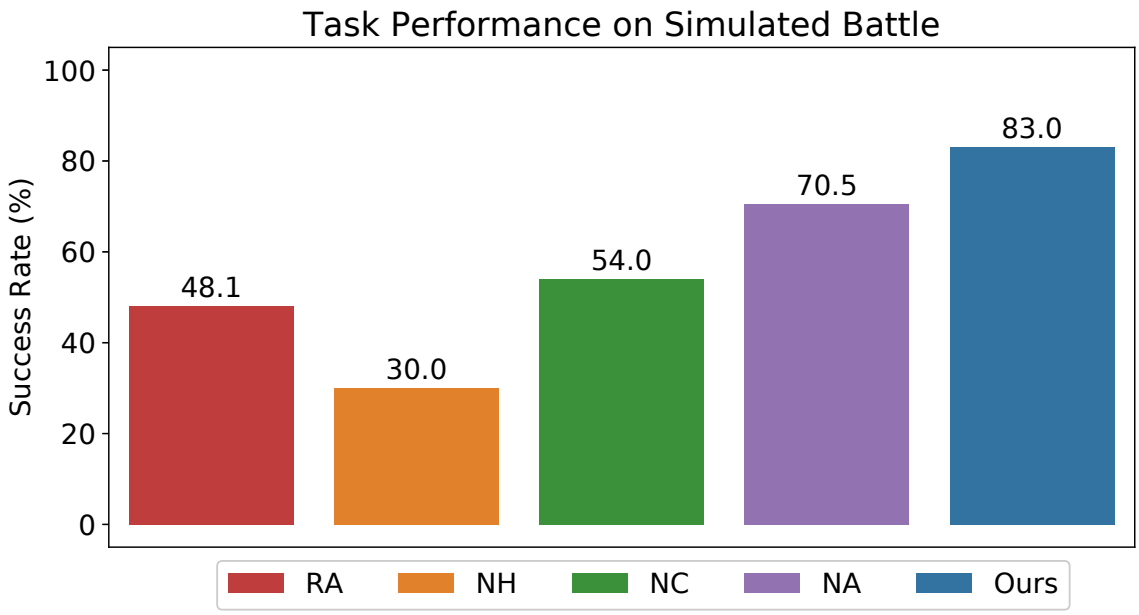


Figure 4.4: Success rate of our approach and that of the baselines on StarCraft battles (higher is better).

against 10 Stalkers and 15 Zealots. We then test each approach on a set of 10 target teams ($\{Q^{(j)}\}_{j=10}^{19}$), each consisting a subset of five total available species. Further details of the experimental design can be found in Appendix B.

Results and discussion

Our approach identified three strategies using Agglomerative Clustering from the 9 demonstrations. Strategy 2 has the highest Att. (A) and DPS (A) which measure the ability of the coalition to attack the enemy from the air while Strategy 3 has no aerial attack power but makes up for it with the highest armor and health capabilities. Strategy 1 has moderate attack and defensive capabilities when compared to 2 and 3. On evaluating the coalitions optimized by the approaches from the inferred strategies, we note that the trends in terms of trait mismatch and robot utilization are similar to those from the numerical experiments.

As shown in Figure 4.3, the RA baseline results in the highest E_{exact} and a non-trivial E_{min}, even when utilizing all the available robots. In contrast, our approach (ours) and all other baselines (NH, NC, and NA) utilize fewer proportion of robots and result in comparable

or lower errors. These observations demonstrate the need for careful task allocation.

We find that our approach (ours) outperforms baselines (NH and NC) in terms of both minimum and exact trait mismatch errors, and that the improvements are statistically significant ($p < 0.01$). Further, our approach is able to do so while utilizing a comparable number of robots. This observation indicates that our approach utilizes the recruited robots more effectively.

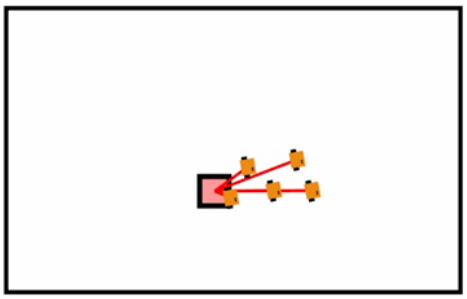
Finally, ignoring abstraction (NA) incurred a considerably larger computational cost (13.22 ± 7.82s) than our approach (1.39 ± 1.17s)[3]. These findings suggest that accounting for heterogeneity, context, and appropriate abstraction results in higher computational efficiency and better satisfaction of task requirements.

On comparing the success rates reported in Figure 4.4, the NH baseline performs worse than all other baselines including RA. This observation points to the fact that ignoring heterogeneity and relying on statistical averages of multi-modal distributions could lead to adverse effects. Further, a potential explanation for the performance of RA baseline could be the existence of additional strategies that were not represented in the demonstrations. Note that, even in such circumstances, our approach is able to select one of the inferred strategies that is best suited for the team.

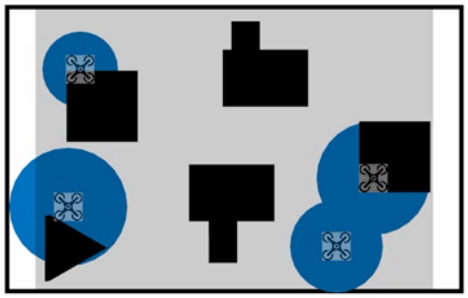
Generally, approaches with lower E_{min} resulted in higher success rates. Our approach results in the highest success rate across all tasks and only resulted in two failures, both due to under-resourced teams. While the approach that ignores abstraction (NA) had the second highest win rate, it is considerably more computationally expensive. In summary, our approach (ours) outperforms all the baselines in terms of both trait satisfaction and task performance without incurring significant computational burden.

[3] See Appendix Appendix D for more details

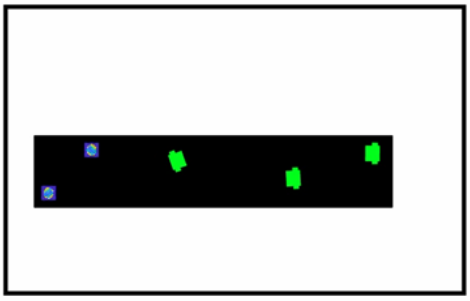
Figure 4.5: We simulate three tasks as shown on Robotarium. Figure 4.5a shows a coalition of ground robots moving debris from a starting location to a goal location. Figure 4.5b shows a coalition of aerial robots searching an urban environment and Figure 4.5c is a simulation of miniature ground robots retrieving objects from a narrow passage.



(a) Move debris.



(b) Search an urban environment.



(c) Retrieve objects from a narrow passage.

4.4.5 Evaluation on Robotarium

In our final set of experiments, we design a multi-robot emergency-response scenario on the Robotarium, a publicly-accessible testbed [51].

Design

We designed the following three tasks for the emergency-response mission:

- *Move debris*: Two to five robots form a coalition and move a piece of debris weigh-

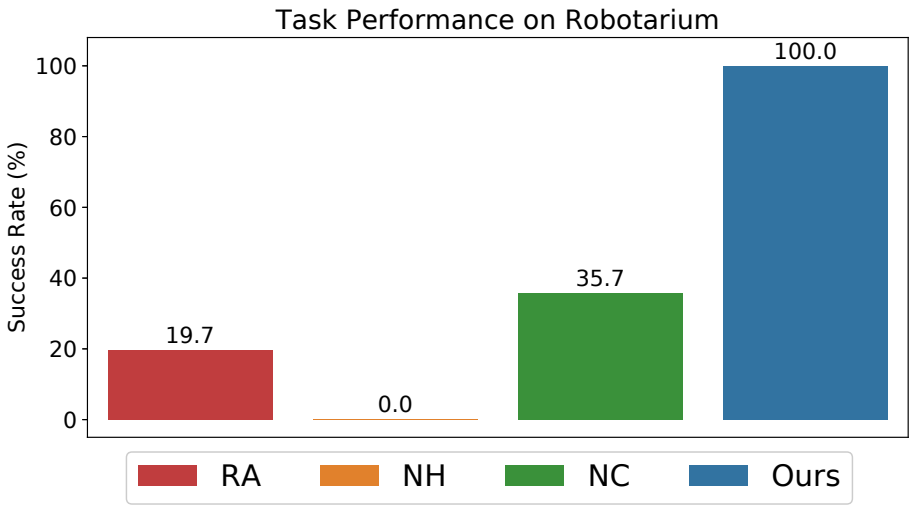


Figure 4.6: Success rate of our approach and that of the baselines on robotarium tasks (higher is better).

ing 50 pounds to a goal location. The capabilities needed for the task are payload capacity and mobility.

- *Search an urban environment*: Two to five robots search an urban environment to look for survivors. The capabilities needed for the task are coverage area and mobility.

- *Retrieve objects from a narrow passage*: Robots are required to navigate and retrieve objects from a narrow passage. The capabilities needed for the task are miniature size, coverage area, and mobility.

The strategies for all three tasks involve the use of either ground robots or aerial robots as shown in Table C.1 of Appendix C.

The test team consists of 5 ground robots, 4 aerial robots, 3 miniature ground robots and 1 miniature aerial robot. We evaluate all the approaches on the test team by computing 100 possible coalition-task pairs for every task.

Results and discussion

On comparing the task performance as shown in Figure 4.6, our approach (ours) consistently assigns teams that satisfy the trait requirements across all tasks, achieving a 100% success rate. The RA baseline fails to assign meaningful coalitions in most cases and results in low success rate (19.7%) indicating that the tasks require careful allocation. Notably, all the coalitions assigned by the NH baseline fail because computing the statistical average of all demonstrations does not result in meaningful trait requirements. Finally, the NC baseline performs better than both NH and RA baselines, but results in a considerably lower success rate (35.7%) compared to our approach, indicating that context-dependent selection of strategies is necessary.

Taken together, the three experiments validate the need for the different components of our framework (ours) and demonstrate that our approach outperforms all baselines (NH, NC, NA, and RA) across all experimental conditions.

CHAPTER 5
LEARNING TO FORM COALITIONS FROM SUB-OPTIMAL DEMONSTRATIONS

In this chapter, we propose an interactive bandit-based approach to learn to form coalitions, given sub-optimal demonstrations from users with respect to their performance on tasks. To motivate this problem, consider two example tasks as battle scenarios on StarCraft II, and let the available species in our team be Zealots, Stalkers, Marines and Marauders. A good assignment to the two battles will utilize minimum resources while eliminating all the opponents, however, what would happen if the users allocate sub-optimal coalitions? We assume that the aggregated traits computed from the agents assigned to each battle has a direct relationship with the obtained scores. We discuss methods in which this relationship between the aggregated traits and the scores are learned by i) using teams and scores from sub-optimal demonstrations (passive) ii) using teams not seen in the demonstrations (active) or iii) a combination of both.

5.1 Problem Statement

Given N_{sub} sub-optimal demonstrations in the form of $\mathscr{D}_s = \{X^{(k)},\ Q^{(k)},\ S^{(k)}\}_{i=1}^{N_{sub}}$ for M tasks where $X^{(k)} \in \mathbb{R}_+^{M\times S}$, $Q^{(k)} \in \mathbb{R}_+^{S\times U}$ and $S^{(k)} \preceq S_{max}$, $S^{(k)} \in \mathbb{R}_+^{M}$, we would like to find an allocation $X^{(l)}$ for a new team $Q^{(l)}$ which maximizes the scores.

In other words, given N_{sub} demonstrations in the form of $\mathscr{D}_s$, we are interested in learning a reward function $\hat{r}_m$ that determines any team's performance on the tasks. The estimate of the scores defined in Equation 3.0.7 are computed using the learned reward function. Our goal is to learn a mapping function from the aggregated traits to scores which will enable the allocation of coalitions capable of achieving performance better than the demonstrators. In the next section, we discuss the details of the learning component of our approach.

5.2 Learning from Sub-Optimal Demonstrations

We approach the problem of learning from sub-optimal demonstrations by treating it as two sub-problems: i) to identify the reward function associated with the assignments ii) to compute the allocation for a new team by either optimizing the learned reward function or optimizing the least squared error between the learned trait requirements represented as $\hat{Y}$ and the aggregated traits of the team.

First, we discuss a passive learning architecture to learn the scoring function directly from demonstrations, for each task separately. In this approach, there is no access to interactions from the environment. Second, we propose a bandit-based active learning architecture which uses interactions from the simulator directly to estimate the trait requirements for tasks. Finally, we discuss how our approach combines both demonstrations and interactions from the simulator. Our objective is to bootstrap learning from sub-optimal demonstrations in the bandit-based framework to achieve large generalizable task performance when new teams need to be allocated.

The learned reward function for any task m is given by $\hat{r}_m : \mathscr{Y} \rightarrow \mathbb{R}^+$, $\forall m$, where $\mathscr{Y}$ represents the space of all possible trait requirement vectors. Note that, $\hat{r}_m$ is an estimate of the ground truth reward distribution defined in Equation 3.0.6.

For both the reward and bandit-based interaction learning, we compute the trait aggregation from the demonstrations as

$$Y^{(l)} = X^{(l)} Q^{(l)} \tag{5.1}$$

where $Y^{(l)} \in \mathbb{R}_+^{M \times U}$ and each row of $Y^{(l)}$ is represented as $\{y_{ml}\}_{m=1}^{M}$ which is the set of traits that the demonstrated allocation made available for each task. In the following sections, we develop the theory of learning evaluation functions and discuss their performance in numerical simulations.

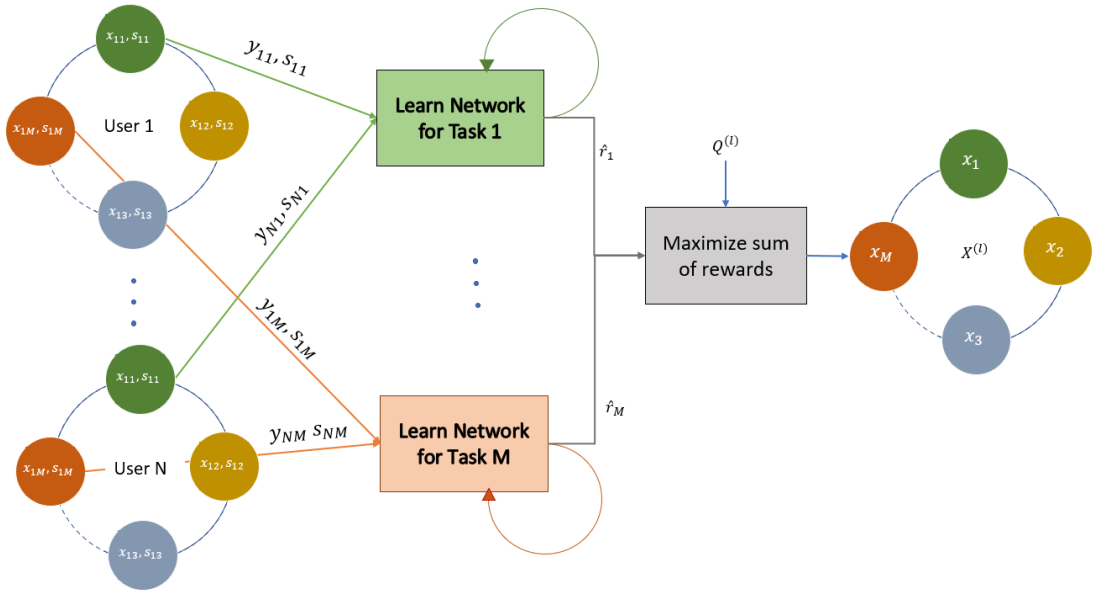


Figure 5.1: The block diagram represents the method for learning the reward function from the demonstrated aggregated traits.

5.3 Reward Learning using Neural Networks

First, we introduce a neural network based reward regression to approximate the reward function using sub-optimal demonstrations as shown in Figure 5.1. This method is inspired from Inverse Reinforcement Learning (IRL) in which, the underlying reward distribution is learned from data. The learned reward function is then used to train RL agents.

Based on principles similar to IRL, we first compute the aggregated traits from the demonstration set $\mathscr{D}_s$ as shown in Equation 5.1. We design a unique reward function network for each task as shown in Figure 5.1. The initial weights of the network are sampled from a normal distribution and trained over all the N_{sub} demonstrations using the aggregated traits as inputs and the corresponding scores as labels. The network learns a function which approximates the relationship between the traits and the scores for each task by minimizing the Mean Squared Error (MSE) between the linear model and the demonstrations.

Note that we use the aggregated traits as opposed to number and types of agents as inputs because, the learned reward functions can generalize when new species, characterized

Algorithm 2: Reward Learning using Neural Networks

Input : $\mathscr{D}_s$ and $Q^{(l)}$
Output: $X^{(l)}$

for `m in range(M)` **do**
    Compute aggregated traits as shown in Equation 5.1, $\forall i \in \{1, \cdots, N_{sub}\}$
    Initialize neural network parameters
    Run k-fold cross-validation with k = 10
    Choose the network with least cross-validation error
    Retrain the network over all the data
    Save the weights
Optimize for $X^{(l)}$ as shown in Equation 5.3 and Equation 5.4
return $X^{(l)}$

by the same set of U traits but with different compositions, are a part of the team.

The output of the network for any task m is given by

$$\hat{s}_m = \hat{r}_m(Q^T x_m) = W_{mo}\phi(W_{mh}(Q^T x_m) + b_{mh}) + b_{mo} \tag{5.2}$$

where ϕ represents the activation function on the nodes, $Q^T x_m$ represents the input vector, W_{mo}, b_{mo} are the weights and biases of the output layer respectively, W_{mh}, b_{mh} are the weights and biases of the input layer respectively, $\hat{r}_m$ is the network function learned for task m and $\hat{s}_m$ is the estimated score associated with $Q^T x_m$. This design is extrapolated for all tasks in the task set $\mathscr{T}$.

In the second part of this method, we use the learned reward functions of all M tasks to compute the allocation for a new team $Q^{(l)}$. To this end, we maximize the sum of the network outputs while constraining the inputs (aggregated traits) to not exceed the total available traits possessed by the agents of our team. Precisely, we solve a constrained maximization problem as follows.

$$X^{(l)} = \arg\max_{X^{(l)}} \sum_m \hat{r}_m(X^{(l)} Q^{(l)}) \tag{5.3}$$

$$s.t.\ X^{(l)T} \cdot 1 \preceq N_a \tag{5.4}$$

subject to the constraints of not allocating more than the available agents in our team where N_a represents the vector of available agents per species. The performance of reward based regression is highly dependent on the amount and quality of the sub-optimal demonstrations available for training. Hence, the learned reward function might not generalize well if there are too few demonstrations to learn from. We use this method as a baseline to compare with the performance of the learning component of our approach discussed in the next section.

5.4 Bootstrapping Bandit-based learning with sub-optimal demonstrations

In this section, we describe our bandit-based active learning framework as shown in Figure 5.2 to bootstrap interactive learning with sub-optimal demonstrations. In essence, we propose our approach which generates a prior distribution of the reward function using sub-optimal demonstrations and actively searches the trait space to fine tune the learned reward function.

We model the reward distribution f_m for any task m as a sample from a Gaussian process (GP) defined on a collection of dependent random variables belonging to the continuous space $\mathscr{Y}$, for every subset of which is multivariate Gaussian distributed in an overall consistent way. A GP is specified by its mean function μ_m and covariance function k_m for task m and defined as follows:

$$f_m : \mathscr{Y} \rightarrow \mathbb{R}_+ \tag{5.5}$$

$$\mu_m(y_m) = \mathbb{E}[f_m(y_m)] \tag{5.6}$$

$$k_m(y_m, \bar{y}_m) = \mathbb{E}[(f_m(y_m) - \mu_m(y_m))(f_m(\bar{y}_m) - \mu_m(\bar{y}_m))] \tag{5.7}$$

where $y_m, \bar{y}_m \in \mathscr{Y}$ are two random variables in the trait vector space.

We define the reward r_{mt} sampled at y_{mt} for task m at iteration t as

$$r_{mt} = f_m(y_{mt}) \tag{5.8}$$

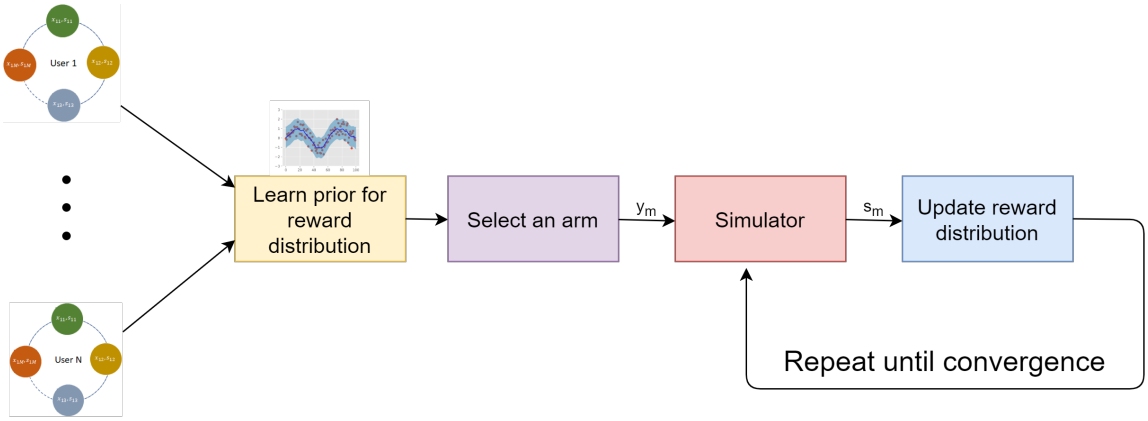


Figure 5.2: The block diagram shows the approach of using demonstrations and bandit-based interactions for learning the trait requirements.

where $\mathscr{Y} \in [0,1]^d$, $y_{mt} \in \mathscr{Y}$ $\forall m$ and $\forall t \in \{1,\cdots,T\}$ with total number of iterations denoted by T. For all the tasks in $\mathscr{T}$, the scores obtained by the coalitions are the rewards.

For a sample $\mathbf{r}_{mT} = [r_{m1},\ r_{m2},\cdots,\ r_{mT}]^T$ at points $\mathbf{A}_{mT} = \{y_{m1},\ y_{m2},\cdots,\ y_{mT}\}$, the posterior of f is a GP distribution again with mean $\mu_{mT}(y_m)$, covariance $k_{mT}(y_m,\overline{y_m})$ and variance $\sigma^2_{mT}(y_m)$:

$$\mu_{mT}(y_m) = \mathbf{k}_{mT}(y_m)^T \mathbf{K}_{mT}^{(T)-1} \mathbf{r}_{mT} \tag{5.9}$$

$$k_{mT}(y_m,\overline{y_m}) = k_m(y_m,\ \overline{y}_m) - \mathbf{k}_{mT}(y_m)^T \mathbf{K}_{mT}^{-1} \mathbf{k}_{mT}(\overline{y_m}), \tag{5.10}$$

$$\sigma^2_{mT}(y_m) = k_{mT}(y_m,y_m) \tag{5.11}$$

where $\mathbf{k}_{mT}(y_m) = [k_{mT}(y_{m1},y_m),\ k_{mT}(y_{m2},y_m),\cdots,\ k_{mT}(y_{mT},y_m)]^T$ and $\mathbf{K}_{mT}$ is the positive definite kernel matrix $[k_{mT}(y_m,\overline{y_m})]_{y_m,\ \overline{y_m}\in\mathbf{A}_{mT}}$.

A key contribution of this method is the idea of treating each aggregated trait vector $y_m \in \mathscr{Y}$ of task m as an "arm" in a multi-armed bandit problem. In this particular instance, the input space of the multi-armed bandit is continuous. However, we note that every point in this continuous space is not "reachable" by the available traits in the team. This is because the traits of any individual agent cannot be divided (see Appendix F for more

details). Inspired from the GP-UCB algorithm [18], we design a continuous arm bandit approach that selects arms based on the Upper Confidence Bound (UCB) rule.

The UCB selection rule states that the arm with the highest confidence bound is selected at each iteration. We apply the UCB rule by selecting the arm that maximizes the sum of the mean and the weighted standard deviation of the reward distribution for task m at each iteration t as shown in the equation below:

$$y_{mt} = \arg\max \mu_{m(t-1)} + \beta_{mt}^{1/2} \sigma_{m(t-1)} \tag{5.12}$$

where y_{mt} is the arm returned by the optimization, $\mu_{m(t-1)},\ \sigma_{m(t-1)}$ are the sample mean and standard deviation of the GP respectively and β_{mt} is a parameter to trade-off between exploration and exploitation.

Finally, we estimate $\hat{Y} \in \mathbb{R}_{+}^{M \times U}$ by selecting the arms that maximizes the reward for all the M tasks and solve the constrained optimization problem to find $X^{(l)}$ as follows:

$$X^{(l)} = \arg\min_{X^{(l)}} \ \|\hat{Y} - X^{(l)} Q^{(l)}\|_2^2 \tag{5.13}$$

$$s.t.\ X^{(l)T} \cdot 1 \preceq N_a \tag{5.14}$$

where $\hat{Y}$ represents the selected task requirements and Equation 5.14 is the constraint that does not allow for allocating more than total available agents in the team.

To summarize, we introduce prior information from sub-optimal demonstrations by considering each demonstration as an arm. The Bayesian update is performed using the score obtained by the coalition formed by that arm (aggregated traits). After N_{sub} iterations, the computed mean and standard deviation of the distribution is used as the GP prior. For the next T iterations, we sample values from the continuous trait space using the UCB selection rule according to Equation 5.12 to find the optimal arm. This process is repeated for every task and the optimal arms found are used to form $\hat{Y}$. Finally, the alloca-

Algorithm 3: Learning trait requirements using GP-UCB

Input : Input Space $\mathscr{Y}$, GP Prior computed on $\mathscr{D}_s$, $Q^{(l)}$
Output: $X^{(l)}$

for $m = 1, 2, \cdots, M$ **do**
  for $t = 1, 2, \cdots, T$ **do**
    Compute optimal β_{mt}
    Choose y_{mt} based on Equation 5.12
    Sample $r_m(y_{mt}) = f_m(y_{mt})$
    Perform Bayesian update to obtain μ_{mt} and σ_{mt}
Optimize for $X^{(l)}$ as shown in Equation 5.13, Equation 5.14
return $X^{(l)}$

tion of agents is computed by solving the least squares problem defined in Equation 5.13 constrained by Equation 5.14.

5.5 Experimental Evaluation

In this section, we evaluate our approach[1] in terms of its ability to learn evaluation functions and form coalitions using detailed numerical simulations. Precisely, we measure our framework's ability to generalize the learned evaluation function and trait requirements to teams with different number of agents, and different types of species.

5.5.1 Baselines

We compare the performance of our approach against the following baselines:

1. **Average Y*:** In this baseline, the sample mean of the aggregated traits of all the demonstrations is computed and assumed to be an estimate of the requirements for tasks.

$$\hat{Y}^* = \sum_{k=1}^{N_{sub}} \frac{X^{(k)} Q^{(k)}}{N_{sub}} \tag{5.15}$$

   The agent assignment is optimized by solving the constrained least squares problem

[1] https://github.com/Ravichandar-Lab/LTAfSD

as follows.

$$X^{(l)} = \arg\min_{X_k} \|\hat{Y}^* - X^{(l)}Q^{(l)}\|_2^2 \tag{5.16}$$

$$s.t.\ X^{(l)T} \cdot 1 \preceq N_a \tag{5.17}$$

2. **Reward Net:** In this baseline, the underlying reward distribution is learned directly from the demonstrations using neural networks and the agent assignment is optimized by maximizing the sum of the rewards across all tasks as shown in Equation 5.3.

3. **Gaussian Process Bandits without prior information (GPUCB):** In this baseline, we implement the GP-UCB algorithm as shown in Figure 5.2 without the "Learn prior for GP" block. We initialize the mean and standard deviation of the Gaussian processes for all tasks to 0. The arms (aggregated trait vectors) for each task, are chosen based on the UCB selection rule and the baseline is run until convergence. The best arms found for all tasks are together considered as the task requirements denoted by $\hat{Y}$. The agent assignment is computed by optimizing the least squares problem as shown in Equation 5.13 and Equation 5.14.

5.5.2 Metrics

1. **Average performance:** The average performance is defined as the sum of the scores achieved by the different coalitions on tasks, divided by the total number of test teams. For W test teams, the average performance (G_p) is given by:

$$G_p = \frac{\sum_W \sum_{m=1}^{M} s_m}{W} \tag{5.18}$$

2. **Iterations to convergence:** The maximum number of iterations taken by the bandit approaches to obtain their highest score is defined as the iterations to convergence.

5.5.3 Numerical Simulations

Design

For learning the scoring function from the demonstrations, we design a neural network for each task. Each network block as shown in Figure 5.1, consists of a single hidden layer with five neurons. The activation function is the sigmoid function with batch size set as 5 and number of epochs set to 1000. The Adam Optimizer is used to compile the model with the loss function as the mean squared error. During training, we use 100 demonstrations of different allocations to two tasks, made from a team consisting of 16 agents of species type 1, 4 agents of type 2, 8 agents of type 3 and 11 agents of type 4. The scores for the allocations are assumed to be a part of the demonstrations. The true scoring function considered for numerical simulation is as follows:

$$s_m = \begin{cases} \frac{1800}{\frac{(\|X^{(l)}Q^{(l)} - Y_1^*\|_2)}{2}} & \text{if } m = 1 \\ \frac{2700}{(\|X^{(l)}Q^{(l)} - Y_2^*\|_2)} & \text{if } m = 2 \end{cases} \tag{5.19}$$

for an assignment matrix $X^{(l)}$ and team characterized by $Q^{(l)}$.

Results and Discussion

For the experiment, we evaluated our approach against the baselines on 100 test teams with different types of species and number of agents per species and found the results to be as shown in Figure 5.3. Our approach is able to achieve the highest average generalization performance of 4.96 while the GPUCB baseline performs the second best with 4.61. The Average Y* method performs poorly with G_p 4.26 because of the fact that computing the average of the aggregated traits may not lead to learning meaningful trait requirements. Reward Net baseline performs better than the Average Y* method with G_p as 4.55, however, it performs worse than both GPUCB and our approach. Instead of looking at G_p, if we

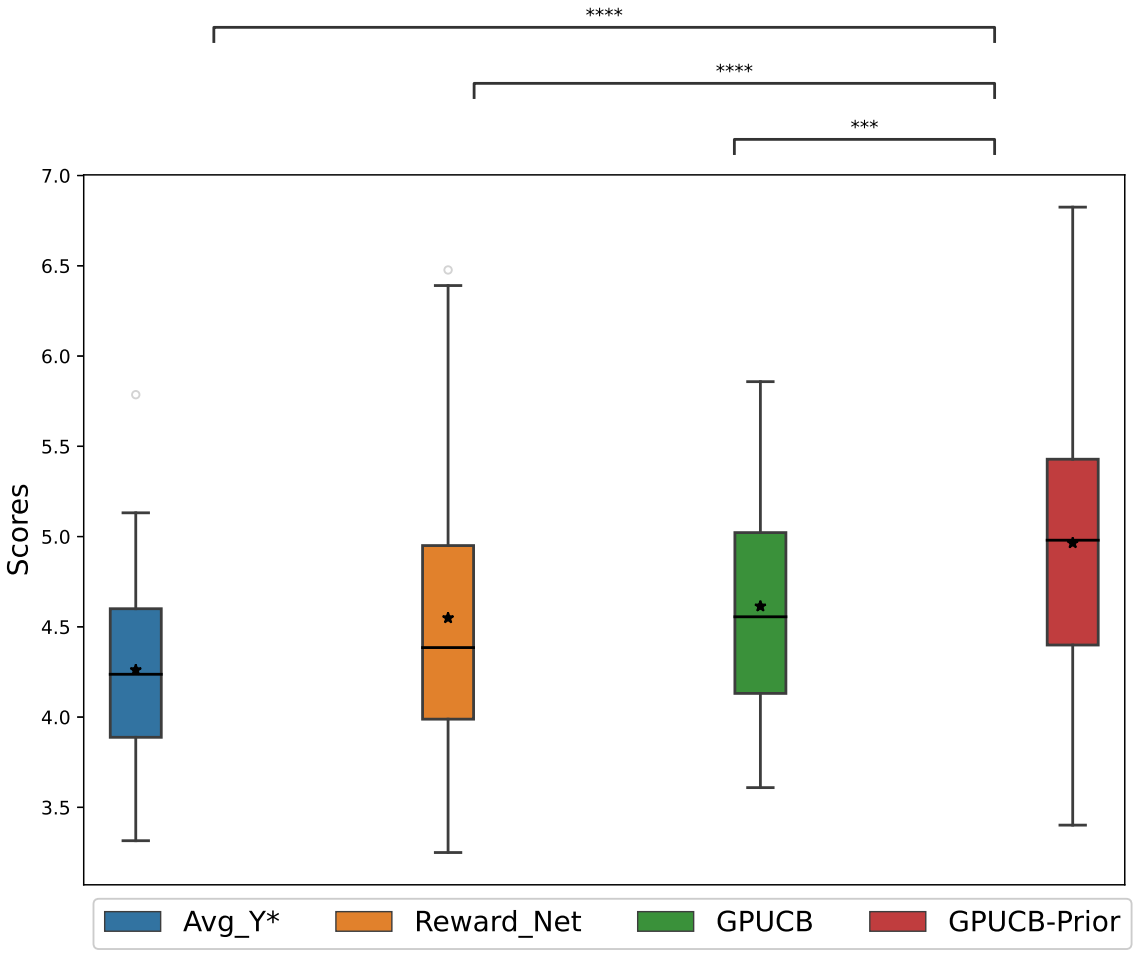


Figure 5.3: Average Generalization Performance of our approach and that of the baselines evaluated using 100 test teams on two numerical simulation tasks (higher is better).

compare the highest score achieved by the approaches, we see that our approach is closely followed by the Reward Net baseline, followed by GPUCB while the Average Y* obtains the least highest score. Overall, this shows evidence that bootstrapping the bandit-based interaction learning framework with demonstrations results in the formation of coalitions achieving the highest performance scores.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

6.1 Conclusion

In the first part, we proposed an approach to learn coalition formation strategies when task requirements are not explicitly known. Our framework was capable of distilling expert demonstrations into *heterogeneous strategies* that capture different, equally-valid trait requirements. Further, our framework was effective in generalizing the inferred strategies to new target teams by considering the capabilities of the specific target team (i.e., context) without requiring additional training. Detailed experiments using numerical simulations, StarCraft II battles, and a multi-robot emergency-response mission reveal that our approach outperforms existing approaches to coalition formation that ignore i) the heterogeneity in strategies, ii) the context of available resources, or iii) appropriate abstraction to extract a small number of strategies. Next, we addressed the research problem of learning from sub-optimal demonstrations by contributing a bandit-based interactive learning framework to bootstrap sub-optimal demonstrations with data from the simulator. The results from numerical simulations show that our approach outperforms baselines that only learn from passive data (sub-optimal demonstrations) or baselines that only learn from active data (bandit-based interaction without sub-optimal demonstrations).

6.2 Future Work

Our first contribution of a resource-aware generalization framework assumes that optimal demonstrations are provided by experts. Additionally, we assume that the different strategies are equally valid solutions to tasks. However, in many cases, the demonstrations from users are sub-optimal with respect to task performance. This will result in poor generaliza-

tion performance as well. In our second contribution, we address the problem of learning from sub-optimal demonstrations. In future, we would like to extend this to incorporate multiple strategies and propose a method to learn from sub-optimal demonstrations when there are equally valid strategies for solving tasks. Finally, we would like to extend our resource-aware generalization framework to learn strategies that will make the system more resilient towards failures. Based on environmental uncertainties, some strategies may be more appropriate for different situations.

The second framework we propose for learning from sub-optimal demonstrations learns a separate reward distribution for each task. As a result, this approach is applicable only to situations where the reward distributions over tasks have a global maximum. Our approach would fail if the reward distributions were monotonically increasing in nature, since all the available agents would be assigned to every task to receive the maximum reward. To conclude, we would like to extend our approach to tackle other kinds of reward distributions by optimizing the learned GPs directly to compute coalitions.

Appendices

APPENDIX A

ADDITIONAL NUMERICAL ANALYSIS

In addition to the numerical experiments reported in subsection 4.4.3, we conducted additional numerical experiments by varying the number of species and the number of tasks. We considered 20 such datasets by varying the number of available species from $\{2, 4, 6, 8, 10\}$ and the number of concurrent tasks from $\{2, 3, 4, 5\}$. We report that the minimum and exact trait mismatch errors follow similar trends as seen in Figure B.1. The baselines considered for this experiment are namely RA, NH and NC. The trait requirements for each task is generated by sampling from a Gaussian mixture model with 3 clusters.

APPENDIX B

DETAILS OF STARCRAFT II EXPERIMENTS

For the design of StarCraft II experiments, we use 4 species from the game, namely Zealots, Stalkers, Marines and Marauders and simulate demonstrations by specifying different combinations of ally teams that lead to wins against the enemy team. The traits of the different species are listed in Table B.1. We test our approaches with an additional species not seen in the demonstrations namely Roach.

Table B.1: Traits of Species for StarCraft II Experiments

Traits	Zealot	Stalker	Marine	Marauder	Roach
Armor	1	1	0	1	1
Health	100	80	45	125	145
Shield	50	80	0	0	16
Att. (G)	8	13	6	5	16
Att. (A)	0	13	6	0	0
DPS (G)	18.6	9.7	9.8	9.3	11.2
DPS (A)	0	9.7	9.8	0	0
Cooldown	0.86	1.34	0.61	1.07	1.43
Speed	3.15	4.13	3.15	3.15	3.15
Range	0	6	5	6	4
Sight	9	10	9	10	9

For the design of traits that are non-cumulative such as Cooldown, we compute the inverse to make it cumulative. The Speed trait threshold is set at 4, which implies that species that have movement speed greater than 4 assume the binary value 1 and it is 0 otherwise.

Table B.2: Inferred Strategies for Battle (10s15z)

Traits	Strategy 1	Strategy 2	Strategy 3
Armor	28.67	0	30
Health	2566.67	2010	3225
Shield	1883.33	0	1050
Att. (G)	304.33	268	213
Att. (A)	195	268	0
DPS (G)	399.7	223.33	474.3
DPS (A)	145.5	437.73	0
Cooldown	27.09	73.22	32.83
Speed	15	0	0
Range	90	223.33	54
Sight	273	402	279

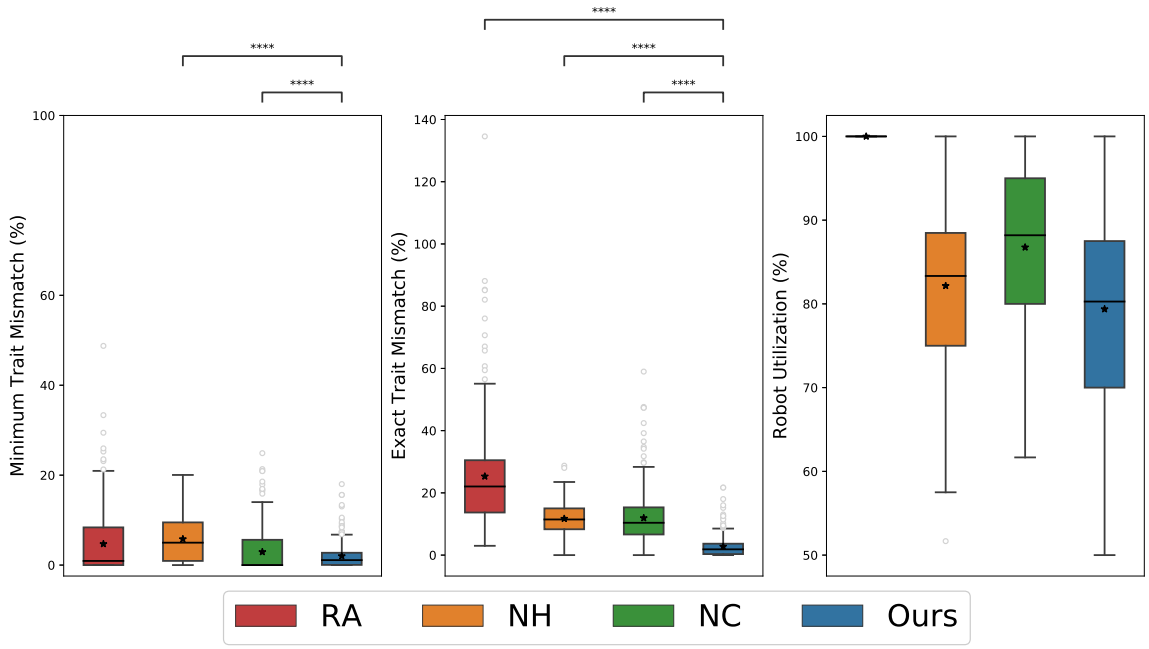


Figure B.1: Performance, as measured by trait satisfaction and robot utilization percentages, of our approach and that of the baselines on 20 datasets (lower is better).

APPENDIX C

DETAILS OF ROBOTARIUM EXPERIMENTS

The strategies for all tasks are mentioned in Table C.1. For task 1, the debris can be moved by either a coalition of ground robots or aerial robots with collective payload capacity meeting the trait requirements. Hence, we have ground robots and aerial robots encoded as separate traits. The total area to be covered to search an urban environment in task 2 is 8 m^2 and can be achieved using either ground or aerial robots. In task 3, either miniature ground robots with higher coverage area or miniature aerial robots with smaller coverage area can be used to retrieve objects.

Table C.1: Inferred Strategies for Tasks on Robotarium

	Coverage Area (m^2)	Ground Robots	Payload Capacity (lb)	Miniature Quality	Aerial Robots
Task 1	0 0	0 5	50 50	0 0	5 0
Task 2	8 8	4 0	0 0	0 0	0 4
Task 3	6 6	3 0	0 0	3 6	0 6

APPENDIX D

COMPUTATIONAL COSTS

Both the numerical simulation and StarCraft II experiments were performed on an Intel i7(4GHz) CPU with 16GB memory. The numerical simulations were tested on 60 test teams (problem instances) to be assigned to three tasks. On StarCraft II, the experiment was performed on 10 target teams to be assigned to four simultaneous battles. In Table D.1, we report the average and standard deviation of the run times in seconds for the different approaches (for solving a single problem instance). In Table D.2, we report the CPU memory space required in mebibyte (MiB) for the different approaches obtained from the memory profiler.

Table D.1: Computation Time for the different approaches

	NH (s)	NC (s)	NA (s)	Ours (s)
Num.	8.68±34.98	33.53±135.18	1067.7±441.5	11.12±26.94
SC II	1.56±3.31	0.21±0.22	13.22±7.82	1.39±1.17

Table D.2: Memory required for the different approaches

	NH (MiB)	NC (MiB)	NA (MiB)	Ours (MiB)
Num.	169.766	172.195	175.977	164.980
SC II	196.012	191.504	203.508	192.254

APPENDIX E

EVALUATION OF THE DESIGN OF REWARD NET

In order to choose the hyperparameters such as batch size and number of epochs for training the neural network of each task, we ran a k-fold cross-validation pipeline with k set as 10. We selected the network parameters resulting in the least cross-validation error as shown in Figure E.1. The sigmoid activation function with adam optimizer and mean squared error loss function resulted in the average cross-validation error of 0.63569036 (0.03512736) for task 1 and 0.43713198 (0.02900095) for task 2 on the training set and 1.96472085 (1.07366996) for task 1 and 1.29608299 (0.59349945) for task 2 on the testing set.

The results shown in Figure E.2 presents a particular instance of predictions from the networks when the demonstrations are split into training and validation data. The scores shown in Figure E.2 are reported based on the performance of the coalitions across two tasks. The X-axis indicates the true scores computed using the numerical simulator and the Y-axis represents the predicted score from the networks. The score obtained by the optimized allocation using the Reward Net baseline is represented by the red dot.

Figure E.1: The cross-validation error on the training and test data for k = 10 is reported for two networks designed for task 1 and task 2.

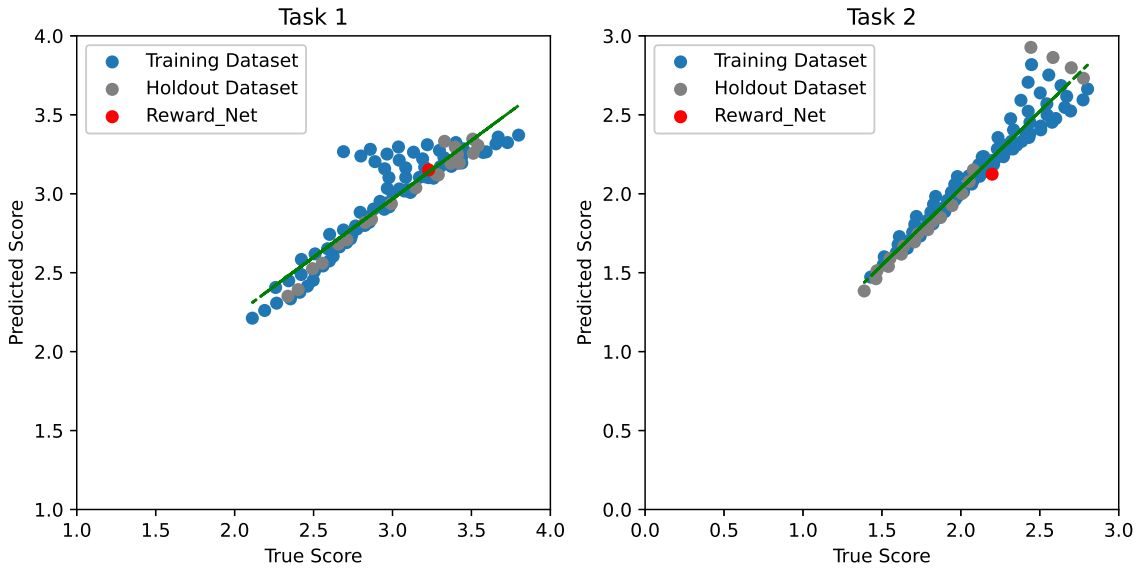


Figure E.2: Predicted score vs true score of Reward Net on demonstrations for task 1 and task 2.

APPENDIX F

IMPLEMENTATION DETAILS OF BOOTSTRAPPING BANDIT-BASED LEARNING WITH SUB-OPTIMAL DEMONSTRATIONS

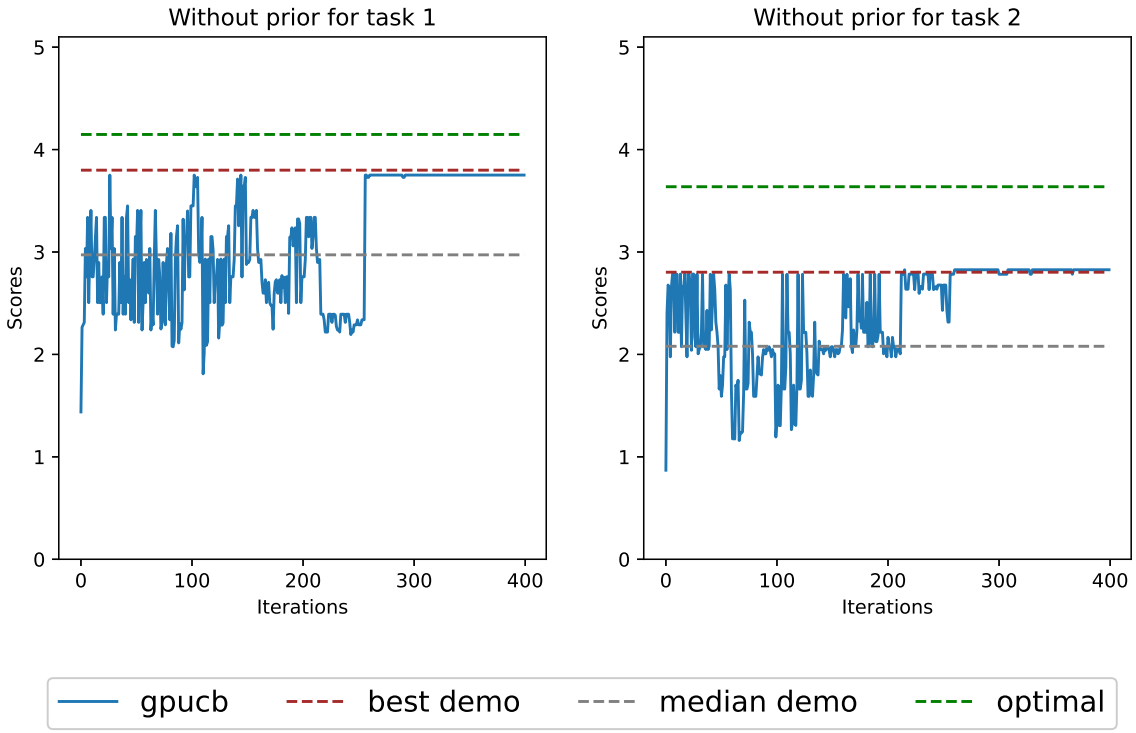
For the bandit-based interaction learning algorithms, we directly sample scores treating the true scoring function like a simulator. Both the bandit-based approaches are run for 400 iterations on each task to identify the best arm for that task. The best arms are together considered as the set of learned trait requirements. The Radial Basis Function is considered as the covariance function for the GPs. Additionally, we evaluate our approach against the baselines to compare the average generalization performance on 100 test teams consisting of different types of species and number of agents. The maximum possible sum of scores that can be achieved on the two tasks is 7.784.

On running the bandit-based approaches for two tasks, we found that on task 1, the GPUCB baseline converges to it's highest score of 3.75 after 256 iterations while our approach (GPUCB-Prior) converges to the highest score of 4.02 after 317 iterations. On task 2, the GPUCB baseline converges to it's highest score of 2.83 after 260 iterations while our approach (GPUCB-Prior) converges to the highest score of 2.96 after 383 iterations as shown in Table F.1.

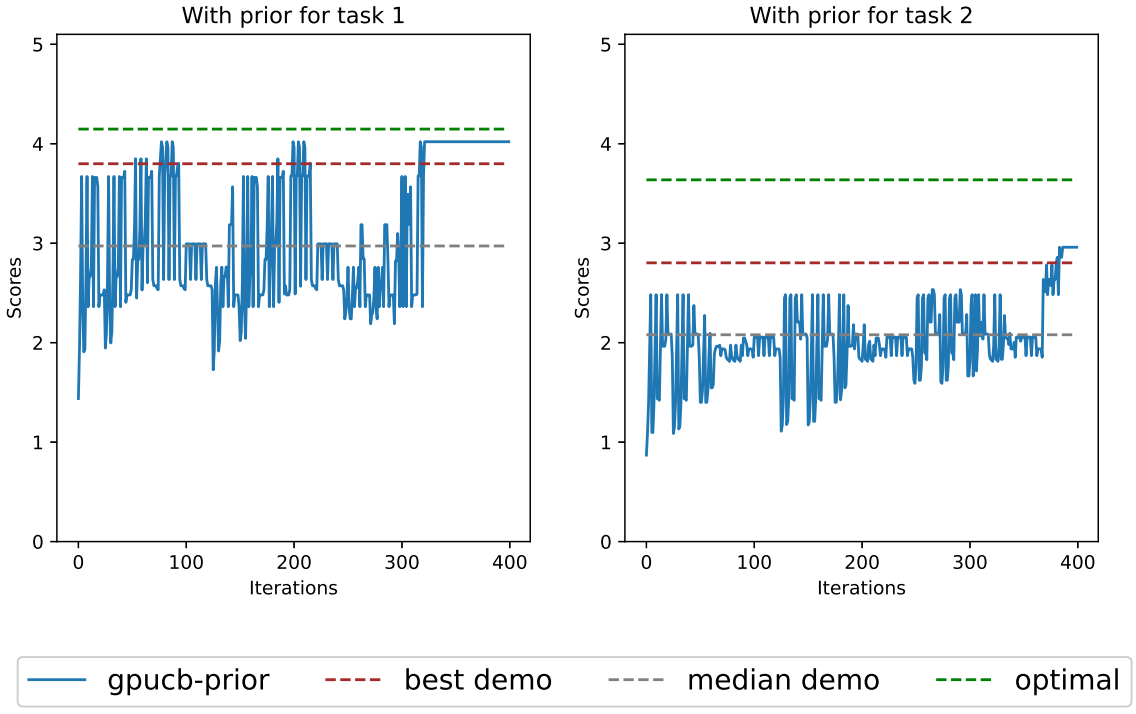
Table F.1: Performance Comparison of GPUCB vs Ours

	Iterations to Conv. (Task 1)	Highest Score (Task1)	Iterations to Conv. (Task 2)	Highest Score (Task 2)
GPUCB	256	3.75	260	2.83
Ours	317	4.02	383	2.96

For the design of numerical simulations, we consider a 4-dimensional trait space. The continuous trait space is represented by a grid of size $4 \times 4 \times 4 \times 4$. Each point in the grid represents an arm (aggregated traits). However, the points in the trait space that are "reachable" are limited by the coalitions that can be formed with the available team. To be



(a) Scores vs Iterations for GPUCB.



(b) Scores vs Iterations for our approach.

Figure F.1: Performance curves of the two bandit-based interaction learning approaches. The top two figures show the number of iterations vs scores for GPUCB on task 1 and task 2. The bottom two figures show the number of iterations vs scores for our approach on task 1 and task 2.

precise, the traits of an individual agent cannot be divided to exactly meet the aggregated traits represented by the arm. Hence, for every arm that is selected by our approach, we compute a coalition that meets the traits as closely as possible by solving a constrained optimization problem as follows:

$$x_m = \arg\min_{x_m} \ \|y_m - Q^T x_m\|_2^2 \tag{F.1}$$

$$s.t.\ x_m \preceq N_a \tag{F.2}$$

where y_m represents the arm selected by our approach, x_m is the optimized coalition, Q is the species-trait matrix and N_a represents the vector of available agents per species. Next, we recompute the aggregated traits from the optimized coalition as:

$$\hat{y}_m = Q^T x_m \tag{F.3}$$

where $\hat{y}_m$ represents the closest set of traits that can be "reached" by the available team. The arm in the grid is replaced by $\hat{y}_m$ before the next iteration. We repeat this process whenever there exists an optimality gap between the selected arm y_m and the "reachable" arm $\hat{y}_m$.

REFERENCES

[1] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International journal of robotics research*, vol. 23, no. 9, pp. 939–954, 2004.

[2] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.

[3] F. Shkurti, A. Xu, M. Meghjani, J. C. G. Higuera, Y. Girdhar, P. Giguere, B. B. Dey, J. Li, A. Kalmbach, C. Prahacs, K. Turgeon, I. Rekleitis, and G. Dudek, "Multi-domain monitoring of marine environments using a heterogeneous robot team," in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE/RSJ, 2012, pp. 1747–1753.

[4] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic uav and ugv system for precision agriculture," *IEEE Transactions on Robotics,*, vol. 32, no. 6, pp. 1498–1511, 2016.

[5] J. Werfel, K. Petersen, and R. Nagpal, "Designing collective behavior in a termite-inspired robot construction team," *Science*, vol. 343, no. 6172, pp. 754–758, 2014.

[6] A. Prorok, M. A. Hsieh, and V. Kumar, "The impact of diversity on optimal control policies for heterogeneous robot swarms," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 346–358, 2017.

[7] H. Ravichandar, K. Shaw, and S. Chernova, "STRATA: A unified framework for task assignments in large teams of heterogeneous agents," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 2, 2020.

[8] R. E. Nisbett and T. D. Wilson, "Telling more than we can know: Verbal reports on mental processes.," *Psychological review*, vol. 84, no. 3, p. 231, 1977.

[9] J. Rieskamp, J. R. Busemeyer, and T. Laine, "How do people learn to allocate resources? comparing two learning theories.," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 29, no. 6, p. 1066, 2003.

[10] J. Rieskamp and P. E. Otto, "Ssl: A theory of how people learn to select strategies.," *Journal of Experimental Psychology: General*, vol. 135, no. 2, p. 207, 2006.

[11] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[12] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, 2016.

[13] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *2013 IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 1331–1336.

[14] J. W. Payne, J. R. Bettman, E. Coupey, and E. J. Johnson, "A constructive process view of decision making: Multiple strategies in judgment and choice," *Acta Psychologica*, vol. 80, no. 1-3, pp. 107–141, 1992.

[15] P. B. Reverdy, V. Srivastava, and N. E. Leonard, "Modeling human decision making in generalized gaussian multiarmed bandits," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 544–571, 2014.

[16] F. Duvallet and A. Stentz, "Imitation learning for task allocation," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2010, pp. 3568–3573.

[17] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[18] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *arXiv preprint arXiv:0912.3995*, 2009.

[19] A. Torreño, E. Onaindia, A. Komenda, and M. Štolba, "Cooperative multi-agent planning: A survey," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, pp. 1–32, 2017.

[20] M. Gombolay, R. Jensen, J. Stigile, S.-H. Son, and J. Shah, "Apprenticeship scheduling: Learning to schedule from human experts," in *AAAI International Joint Conferences on Artificial Intelligence*, 2016.

[21] O. Shehory and S. Kraus, "Task allocation via coalition formation among autonomous agents," in *IJCAI*, Citeseer, 1995, pp. 655–661.

[22] L. Vig and J. A. Adams, "Multi-robot coalition formation," *IEEE transactions on robotics*, vol. 22, no. 4, pp. 637–649, 2006.

[23] F. Afghah, M. Zaeri-Amirani, A. Razi, J. Chakareski, and E. Bentley, "A coalition formation approach to coordinated task allocation in heterogeneous uav networks," in *2018 Annual American Control Conference (ACC)*, IEEE, 2018, pp. 5968–5975.

[24] Y. Zhang and L. E. Parker, "Considering inter-task resource constraints in task allocation," *Autonomous Agents and Multi-Agent Systems*, vol. 26, no. 3, pp. 389–419, 2013.

[25] J. Guerrero and G. Oliver, "Multi-robot task allocation strategies using auction-like mechanisms," *Artificial Research and Development in Frontiers in Artificial Intelligence and Applications*, vol. 100, pp. 111–122, 2003.

[26] L. Lin and Z. Zheng, "Combinatorial bids based multi-robot task allocation method," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, IEEE, 2005, pp. 1145–1150.

[27] B. Xie, S. Chen, J. Chen, and L. Shen, "A mutual-selecting market-based mechanism for dynamic coalition formation," *International Journal of Advanced Robotic Systems*, vol. 15, no. 1, p. 1 729 881 418 755 840, 2018.

[28] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," in *Advances in Neural Information Processing Systems*, 2017, pp. 3812–3822.

[29] A. Tamar, K. Rohanimanesh, Y. Chow, C. Vigorito, B. Goodrich, M. Kahane, and D. Pridmore, "Imitation learning from visual data with multiple intentions.," in *ICLR (Poster)*, 2018.

[30] C. Dimitrakakis and C. A. Rothkopf, "Bayesian multitask inverse reinforcement learning," in *European workshop on reinforcement learning*, Springer, 2011, pp. 273–284.

[31] S. Nikolaidis, R. Ramakrishnan, K. Gu, and J. Shah, "Efficient model learning from joint-action demonstrations for human-robot collaborative tasks," in *2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, IEEE, 2015, pp. 189–196.

[32] R. Paleja and M. Gombolay, "Heterogeneous learning from demonstration," in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, IEEE, 2019, pp. 730–732.

[33] L. Chen, R. Paleja, M. Ghuy, and M. Gombolay, "Joint goal and strategy inference across heterogeneous demonstrators via reward network distillation," in *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, 2020, pp. 659–668.

[34] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous agents and multi-agent systems*, vol. 11, no. 3, pp. 387–434, 2005.

[35] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*, Springer, 2017, pp. 66–83.

[36] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, 2017, pp. 6379–6390.

[37] N. Carion, N. Usunier, G. Synnaeve, and A. Lazaric, "A structured prediction approach for generalization in cooperative multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 8130–8140.

[38] J. Turner, Q. Meng, G. Schaefer, and A. Soltoggio, "Distributed strategy adaptation with a prediction function in multi-agent task allocation," in *AAMAS*, 2018.

[39] G. Chalkiadakis and C. Boutilier, "Sequentially optimal repeated coalition formation under uncertainty," *Autonomous Agents and Multi-Agent Systems*, vol. 24, no. 3, pp. 441–484, 2012.

[40] L. Cui, X. Zhao, L. Liu, H. Yu, and Y. Miao, "Learning complex crowdsourcing task allocation strategies from humans," in *Proceedings of the 2nd International Conference on Crowd Science and Engineering*, 2017, pp. 33–37.

[41] M. E. Taylor, H. B. Suay, and S. Chernova, "Integrating reinforcement learning with human demonstrations of varying ability," in *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, Citeseer, 2011, pp. 617–624.

[42] Y. Li, I. Kash, and K. Hofmann, "Learning good policies from suboptimal demonstrations," in *14th European Workshop on Reinforcement Learning (EWRL 2018)*, vol. 2, 2018.

[43] Y. Gao, H. Xu, J. Lin, F. Yu, S. Levine, and T. Darrell, "Reinforcement learning from imperfect demonstrations," *arXiv preprint arXiv:1802.05313*, 2018.

[44] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg, "Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3612–3619, 2020.

[45] B. Kim, A.-m. Farahmand, J. Pineau, and D. Precup, "Learning from limited demonstrations," in *Advances in Neural Information Processing Systems*, 2013, pp. 2859–2867.

[46] D. S. Brown, W. Goo, and S. Niekum, "Better-than-demonstrator imitation learning via automatically-ranked demonstrations," in *Proceedings of 3rd Conference on Robot Learning*, 2020, pp. 330–359.

[47] J. Le Ny, M. Dahleh, and E. Feron, "Multi-agent task assignment in the bandit framework," in *Proceedings of the 45th IEEE Conference on Decision and Control*, IEEE, 2006, pp. 5281–5286.

[48] A. Rangi and M. Franceschetti, "Multi-armed bandit algorithms for crowdsourcing systems with online estimation of workers' ability," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 1345–1352.

[49] S. Shahrampour, A. Rakhlin, and A. Jadbabaie, "Multi-armed bandits in multi-agent networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2017, pp. 2786–2790.

[50] Y. Wang, I. I. Hussein, and A. Hera, "Evolutionary task assignment in distributed multi-agent networks with local interactions," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2010, pp. 4749–4754.

[51] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.

[52] M. Samvelyan, T. Rashid, C. S. de Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson, "The starcraft multi-agent challenge," *arXiv preprint arXiv:1902.04043*, 2019.